LE910Cx
Software User Guide

1VV0301556 Rev. 1 – 2019-01-31

TELIT
TECHNICAL
DOCUMENTATION

[01.2019]

SPECIFICATIONS ARE SUBJECT TO CHANGE WITHOUT NOTICE

NOTICES LIST

While reasonable efforts have been made to assure the accuracy of this document, Telit assumes no liability resulting from any inaccuracies or omissions in this document, or from use of the information obtained herein. The information in this document has been carefully checked and is believed to be reliable. However, no responsibility is assumed for inaccuracies or omissions. Telit reserves the right to make changes to any products described herein and reserves the right to revise this document and to make changes from time to time in content hereof with no obligation to notify any person of revisions or changes. Telit does not assume any liability arising out of the application or use of any product, software, or circuit described herein; neither does it convey license under its patent rights or the rights of others.

It is possible that this publication may contain references to, or information about Telit products (machines and programs), programming, or services that are not announced in your country. Such references or information must not be construed to mean that Telit intends to announce such Telit products, programming, or services in your country.

COPYRIGHTS

This instruction manual and the Telit products described in this instruction manual may be, include or describe copyrighted Telit material, such as computer programs stored in semiconductor memories or other media. Laws in the Italy and other countries preserve for Telit and its licensors certain exclusive rights for copyrighted material, including the exclusive right to copy, reproduce in any form, distribute and make derivative works of the copyrighted material. Accordingly, any copyrighted material of Telit and its licensors contained herein or in the Telit products described in this instruction manual may not be copied, reproduced, distributed, merged or modified in any manner without the express written permission of Telit. Furthermore, the purchase of Telit products shall not be deemed to grant either directly or by implication, estoppel, or otherwise, any license under the copyrights, patents or patent applications of Telit, as arises by operation of law in the sale of a product.

COMPUTER SOFTWARE COPYRIGHTS

The Telit and 3rd Party supplied Software (SW) products described in this instruction manual may include copyrighted Telit and other 3rd Party supplied computer programs stored in semiconductor memories or other media. Laws in the Italy and other countries preserve for Telit and other 3rd Party supplied SW certain exclusive rights for copyrighted computer programs, including the exclusive right to copy or reproduce in any form the copyrighted computer program. Accordingly, any copyrighted Telit or other 3rd Party supplied SW computer programs contained in the Telit products described in this instruction manual may not be copied (reverse engineered) or reproduced in any manner without the express written permission of Telit or the 3rd Party SW supplier. Furthermore, the purchase of Telit products shall not be deemed to grant either directly or by implication, estoppel, or otherwise, any license under the copyrights, patents or patent applications of Telit or other 3rd Party supplied SW, except for the normal non-exclusive, royalty free license to use that arises by operation of law in the sale of a product.

USAGE AND DISCLOSURE RESTRICTIONS

I. License Agreements

The software described in this document is the property of Telit and its licensors. It is furnished by express license agreement only and may be used only in accordance with the terms of such an agreement.

II. Copyrighted Materials

Software and documentation are copyrighted materials. Making unauthorized copies is prohibited by law. No part of the software or documentation may be reproduced, transmitted, transcribed, stored in a retrieval system, or translated into any language or computer language, in any form or by any means, without prior written permission of Telit

III. High Risk Materials

Components, units, or third-party products used in the product described herein are NOT fault-tolerant and are NOT designed, manufactured, or intended for use as on-line control equipment in the following hazardous environments requiring fail-safe controls: the operation of Nuclear Facilities, Aircraft Navigation or Aircraft Communication Systems, Air Traffic Control, Life Support, or Weapons Systems (High Risk Activities"). Telit and its supplier(s) specifically disclaim any expressed or implied warranty of fitness for such High Risk Activities.

IV. Trademarks

TELIT and the Stylized T Logo are registered in Trademark Office. All other product or service names are the property of their respective owners.

V. Third Party Rights

The software may include Third Party Right software. In this case you agree to comply with all terms and conditions imposed on you in respect of such separate software. In addition to Third Party Terms, the disclaimer of warranty and limitation of liability provisions in this License shall apply to the Third Party Right software.

TELIT HEREBY DISCLAIMS ANY AND ALL WARRANTIES EXPRESS OR IMPLIED FROM ANY THIRD PARTIES REGARDING ANY SEPARATE FILES, ANY THIRD PARTY MATERIALS INCLUDED IN THE SOFTWARE, ANY THIRD PARTY MATERIALS FROM WHICH THE SOFTWARE IS DERIVED (COLLECTIVELY “OTHER CODE”), AND THE USE OF ANY OR ALL THE OTHER CODE IN CONNECTION WITH THE SOFTWARE, INCLUDING (WITHOUT LIMITATION) ANY WARRANTIES OF SATISFACTORY QUALITY OR FITNESS FOR A PARTICULAR PURPOSE.

NO THIRD PARTY LICENSORS OF OTHER CODE SHALL HAVE ANY LIABILITY FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING WITHOUT LIMITATION LOST PROFITS), HOWEVER CAUSED AND WHETHER MADE UNDER CONTRACT, TORT OR OTHER LEGAL THEORY, ARISING IN ANY WAY OUT OF THE USE OR DISTRIBUTION OF THE OTHER CODE OR THE EXERCISE OF ANY RIGHTS GRANTED UNDER EITHER OR BOTH THIS LICENSE AND THE LEGAL TERMS APPLICABLE TO ANY SEPARATE FILES, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

APPLICABILITY TABLE

PRODUCTS

	SW Versions	Modules
LE910 SERIES	25.20.xxx	4G

CONTENTS

NOTICES LIST **2**

COPYRIGHTS **2**

COMPUTER SOFTWARE COPYRIGHTS **2**

USAGE AND DISCLOSURE RESTRICTIONS **3**

APPLICABILITY TABLE **4**

CONTENTS **5**

1. INTRODUCTION **7**

1.1. Scope 7

1.2. Audience 7

1.3. Contact Information, Support 7

1.4. Text Conventions 9

1.5. Related Documents 10

2. HIGH LEVEL SW ARCHITECTURE **11**

3. FUNCTIONAL DESCRIPTION **12**

3.1. General Functionality and Main Features 12

3.2. Application system overview 12

3.2.1. Memory configuration 13

3.2.2. RAM memory 13

3.2.3. Customer application – Storage & configuration 13

3.2.4. Power up time 14

3.2.5. Location Subsystem 14

3.2.6. Application development environment 15

3.2.7. Random number generator 15

3.2.8. SPI 15

3.2.9. GPIO-Keys 16

3.2.10. Serial interfaces 17

3.2.11. Audio 19

3.2.12. UART 21

3.2.13. USB Interface 21

3.2.14. RTC 22

3.2.15. Time Services 22

3.2.16. Data connection 24

3.2.17. WatchDog 24

3.2.18. Power Management 27
3.2.19. Performance build 30
3.3. Basic Operations 31
3.3.1. Command Syntax 31
3.3.2. Command Response Timeout 31
3.3.3. Basic AT Commands 34
3.3.4. RAT and Band Selection 35
3.3.5. SIM/USIM Management 35
3.4. Advanced Operations 41
3.4.1. SMS Management 41
3.4.2. GNSS Management 54
3.5. Packet Switched Data Operations 67
3.5.1. USB Tethering Connection 67
3.5.2. Socket AT Commands 68
3.5.3. SSL AT Commands 72
3.5.4. HTTP AT Commands 76
3.5.5. FTP AT Commands 78
3.5.6. Email AT Commands 82
3.5.7. IOT Platform AT Commands 83

4. PERFORMANCE MEASUREMENTS 86
4.1. Interrupt latencies 86
4.2. Memory bandwidth & Latencies 87

5. SERVICE AND FIRMWARE UPDATE 90
5.1. Firmware Update 90
5.1.1. TFI update 90
5.1.2. XFP update 91

6. GLOSSARY AND ACRONYMS 95

7. DOCUMENT HISTORY 97

1. INTRODUCTION

1.1. Scope

The aim of this document is to introduce Telit LE910Cx module as well as present possible and recommended Software solutions useful for developing a product based on the LE910Cx module. All the features and solutions detailed are applicable to all LE910Cx variants, where “LE910Cx” refers to the variants listed in the applicability table.
If a specific feature is applicable to a specific product, it will be clearly highlighted.

The description text “LE910Cx” refers to all modules listed in the APPLICABILITY TABLE 1.

In this document all the basic functions of a wireless module will be taken into account; for each one of them a valid hardware solution will be suggested and usually incorrect solutions and common errors to be avoided will be highlighted. Obviously this document cannot embrace every hardware solution or every product that may be designed. Obviously avoiding invalid solutions must be considered as mandatory. Whereas the suggested hardware configurations need not be considered mandatory, the information given should be used as a guide and a starting point for properly developing your product with the Telit LE910Cx module.

The integration of the GSM/GPRS/EGPRS/WCDMA/HSPA+/LTE LE910Cx cellular module within user application must be done according to the design rules described in this manual.

The information presented in this document is believed to be accurate and reliable. However, no responsibility is assumed by Telit Communication S.p.A. for its use, such as any infringement of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent rights of Telit Communication S.p.A. other than for circuitry embodied in Telit products. This document is subject to change without notice.

1.2. Audience

This document is intended for Telit customers, especially system integrators, about to implement their applications using our LE910Cx module.

1.3. Contact Information, Support

For general contact, technical support services, technical questions and report documentation errors contact Telit Technical Support at:

- TS-EMEA@telit.com

- TS-AMERICAS@telit.com
- TS-APAC@telit.com
- TS-SRD@telit.com (for Short Range Devices)

Alternatively, use:
http://www.telit.com/support

For detailed information about where you can buy the Telit modules or for recommendations on accessories and components visit:
http://www.telit.com

Our aim is to make this guide as helpful as possible. Keep us informed of your comments and suggestions for improvements.
Telit appreciates feedback from the users of our information.

1.4. Text Conventions

Danger – This information MUST be followed or catastrophic equipment failure or bodily injury may occur.

Caution or Warning – Alerts the user to important points about integrating the module, if these points are not followed, the module and end user equipment may fail or malfunction.

Tip or Information – Provides advice and suggestions that may be useful when integrating the module.

All dates are in ISO 8601 format, i.e. YYYY-MM-DD.

1.5. Related Documents

[1] AT Commands Reference Guide, 80000ST10025a
[2] Refer to the specific "Telit Product Description" document
[3] Refer to the specific "Telit Hardware User Guide" document
[4] IP Easy User Guide, 80000ST10028A
[5] ETSI GSM 07.07, 27.07
[6] EVK2 User Guide, 1vv0300704
[7] ETSI GSM 03.38, 23.038
[8] /
[9] Device Requirements AT&T, Document Number 13340
[10] Telit 3G Modules Ports Arrangements User Guide, 1vv0300971
[11] Enhanced JDR Technical Note, 30353NT11086A
[12] ITU-T Recommendation E.164
[13] ETSI GSM 11.11, 51.011, 31.101, 31.102
[14] ITU-T Recommendation V.24
[15] /
[16] ETSI GSM 11.14, 51.014
[17] Telit 3G Modules AT Commands Reference Guide, 80378ST10091A
[18] Audio Setting Application Note, 80000NT10007A
[19] ETSI GSM 27.005
[20] Telit's Easy Scan User Guide, 1vv0300972
[21] Jamming Detection – HE910 Series Application Note, 80000NT11408A
[22] GE910 Series Ports Arrangements User Guide, 1vv0301049
[23] IP Easy User Guide Application Note, 80000ST10028A
[24] Virtual Serial Device Application Note, 80000NT10045A
[25] NCM Protocol User Guide, 1vv0301246
[26] Telit LE910 V2 Series AT Commands Reference Guide, 80446ST10707A

2. HIGH LEVEL SW ARCHITECTURE

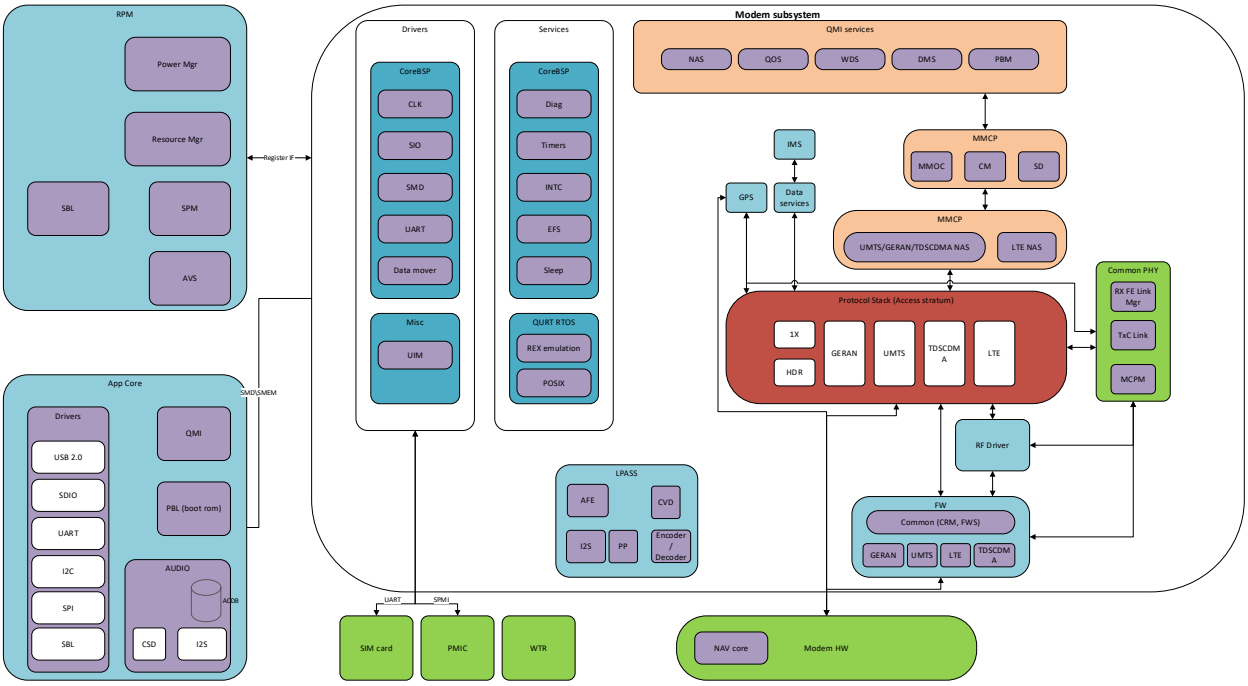


Figure 1 : General System Architecture

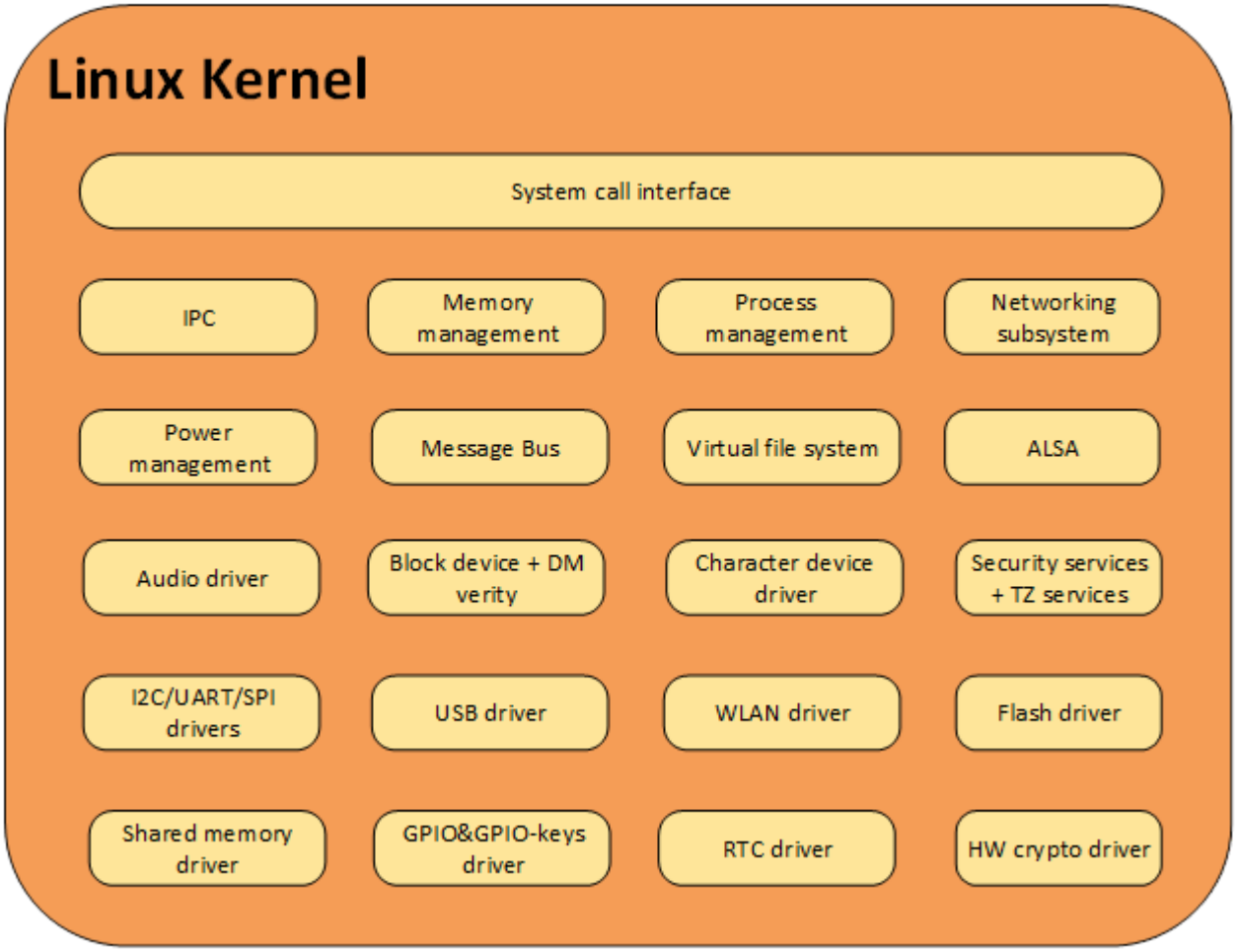


Figure 2: Linux Kernel components

3. FUNCTIONAL DESCRIPTION

3.1. General Functionality and Main Features

The LE910Cx family of cellular modules feature LTE and multi-RAT modem together with an on-chip powerful application processor and a rich set of interfaces.

The major functions and features are listed below:

- Multi RAT cellular modem for voice and data communication
  - LTE FDD/TDD Cat4 (150/50Mbps DL/UL).
  - GSM/GPRS/EDGE
  - WCDMA up to DC HSPA+ Rel. 9
  - Support for SIM profile switching
- Digital audio and analog audio codec
- Application processor to run customer application code
  - 1.2 GHz Cortex-A7 with Linux version 3.18
  - Flash + DDR are large enough to allow for customer's own software applications
- High speed serial interfaces:
  - USB, HSIC
- Tools for firmware update (TFI)
- Stream download protocol (SDL)
- FOTA (Legacy AT FOTA)
- SGMII (optional) for external Ethernet transceiver
- SDIO for (optional) external Wi-Fi transceiver

Note: The LE910C1-SV/LE910C1-SA/LE910C1-ST don't support Wi-Fi.

3.2. Application system overview

The Application Processor is a 32 bit ARM Cortex-A7 up to 1.2GHz running the Linux operating system. The following software is pre-integrated and is running on the application processor:

- 32 bit Cortex-A7@1.2GHz running the Linux kernel 3.18.
- Telit Unified AT command set, backward compatible with LE920, which is the main control interface to offer features by the non-application-enabled variant, with the following:
  - Hayes standard AT command set
  - ETSI GSM 07.07 specific AT command and GPRS specific commands.
  - ETSI GSM 07.05 specific AT commands for SMS (Short Message Service) and CBS (Cell Broadcast Service)
  - Control of pre-integrated Firmware Update Agent (RedBend)
  - Antenna diagnostics
- Firmware Over-The-Air (FOTA) update supporting selective update. Backward compatible with LE910.
- Operator specific Device management client, backward compatible with LE910
- SPI device driver for user space access of the SPI device, including slave to master interrupt, backward compatible with LE910
- GPIO interrupts driver for user space to listen to interrupts on selected user GPIOs, backward compatible with LE910
- 2G/3G/4G and GNSS jamming detection
- Audio subsystem, backward compatible with LE910
  - PCM digital audio IO
  - Limited support for DTMF detection.
- FTM support, backward compatible with LE910
- Vocoder support and processing

- GSM vocoders (EFR/HR/FR), all rates
- AMR-NB, AMR-WB, all rates
- VoLTE
- Configurable noise suppressor, echo canceller and processing chain

- Pre-integrated Wi-Fi driver via SDIO (QualcommA6574)
  - USB for point to point connection (virtual COM)

3.2.1. Memory configuration

The LE910Cx memory configuration is as follows:

- 256Mbyte Flash
- 256Mbyte DDR/128Mbyte DDR

3.2.1.1.1. FS images

- LE910Cx-NF/EU/AP/LA

Root FS – Telit Linux RO root file system

Available memory : about 7 MB is for customer application

- LE910C1-SV/SA/ST

Root FS – Telit Linux RW root file system

3.2.2. RAM memory

- LE910Cx-NF/EU/AP/LA

256Mbyte DDR, of which ~60MB will be available for customers' application usage.

- LE910C1-SV/SA/ST

128Mbyte DDR, there is no available for customers' application usage due to small memory.

No support Telit AppZone Linux with OMADM.

3.2.3. Customer application – Storage & configuration

Using the Telit SDK, the customer application would be installed directly into the USRFS (/data is the mountpoint).

The customer application can also be linked to the powerup process onto predefined hookpoints:

- /data/oem_earlystart.sh – This is at order **38** of the rcS (S is for Single user scripts are run first)
- /data/oemstart.sh - This is at order **43** of the rc5 (5 is for multi user scripts)

- /data/oem_poststart.sh - This is at order **99** of the rc5

Telit rootfs is RO, hence an application cannot be installed into the etc. The above method allows the application to run at powerup. Selecting one of the above methods for application installation should be made based on when the customer wants the application to run during powerup (early, normal and post).

The above scripts should link to the application binary for execution (at least one of them). The installation method as well as build/installation tools should be covered by the Telit LE910C1 SDK document.

Configuration files are stored in three main areas:

1. Telit Linux RO FS. The configuration stored there (mainly in /etc) cannot be changed.
2. Telit RW USER FS (/data). The configuration stored there can be changed by customer application. Examples are hosts, iproute, wlan etc. Configuration stored there are persistent, i.e. written to the flash.
3. Telit RW RAM disk (/var/run). This is a FS mounted on the RAM directly, i.e. not persistent. Examples are DNS, mobile AP and firewall configuration.

Customer applications are installed onto the usrfs storage (/data). This is a RW mountpoint.

The customer application will be automatically linked to the powerup process via predefined scripts in the /data:

- /data/oem_earlystart.sh – This is at order **38** of the rcS (S is for Single user scripts are run first)
- /data/oemstart.sh - This is at order **43** of the rc5 (5 is for multi user scripts)
- /data/oem_poststart.sh - This is at order **99** of the rc5

The above scripts should be then linked to any customer application that needs to automatically run during the powerup process. Note the order of the scripts, rcS scripts runs first on a Linux machine, rc5 script runs after (43 is first then 99).

3.2.4. Power up time

The following measurements were taken using a special perf build.

Non-secured device:

1. Entering kernel: 0.712747
2. Entering user space: 0.742889
3. Entering customer application: 11.046918
4. Modem out of reset: 13.774010

3.2.5. Location Subsystem

The following key features are offered by the Location subsystem:

- Support for GPS, GLONASS, BeiDou/Compass Phase II, Galileo and QZSS
- Supports following Satellite Based Augmentation Systems(SBAS): WAAS, EGNOS, MSAS (only tracked for cross correlation improvement)

- Receiver Autonomous Integrity Monitoring (RAIM) & Fault Detection and Exclusion (FDE) support, internal in the receiver.
- Support of assistance data (Ephemerides, location, time...) provided by customer application to ensure faster Time To First Fix (TTFF) through SUPL and LTO injection
- Periodic pulse output for synchronization to GPS system clock
- NMEA-0183 output on USB

3.2.6. Application development environment

Please refer to the Telit AppZone Linux documentation in the link, below

https://s3.amazonaws.com/site_support/Telit/AppZone-SDK/v4/AppZone_Guide/az-linux-user-guide/index.html

---

Telit AppZone Linux is not available on LE910C1-SA,LE910C1-ST,LE910C1-SV products ,which have small memory,128Mbyte DDR and there is no RAM space for customer application.

3.2.7. Random number generator

The LE910Cx RNG is based on FIPS-140-2 PRNG (aka hw_drbg), seeded with QC designed hw entropy unit

consisting of the ring oscillator (RO) noise source.

There are several Linux devices to generate random numbers (under the /dev node):

1. hw_random – This is an HW random number generator, this is the preferred device to get random data from.
2. random – This is, in most cases, a SW random generator.
   Linux kernel itself (on latest kernel versions >= msm-3.18) adds HW random data to /dev/random if randomness is not sufficient from SW RNG. Random will block if no sufficient randomness is built up.
3. urandom – This device doesn't care if not enough randomness exists and is not recommended for use unless the quality of the RNG is not in concern (this device will probably work faster).

The number of entropies used for the RNG can be checked, and modified, with the following sysfs:

/sys/module/rng_core/parameters/current_quality

Max value is 1024.

An example for reading random bytes from the hw_random:

3.2.8. SPI

The LE910Cx includes a Serial Peripheral Interface (SPI) bus supporting Master mode at 50 MHz (max)

Programmable data bits (4 to 32 bits)

Programmable spacing between byte/word transfers (SPI_CS_N WAIT)

Programmable transfer length (number of bytes transferred within each SS/CS assertion)

TGPIO 1-6 (customer allocated GPIO) can be configured as an interrupt source of an SPI master device. This allows an SPI slave device to notify the SPI master device of data being transferred.

A new module, SPIDEV, is introduced, in which will accept a TGPIO parameter (1-6) to be used as an interrupt from the client to the master.

The SPI uses BLSP (board low speed peripheral).

LE910Cx has one BLSP core.

The BLSP includes a UART and QUP (Qualcomm universal peripheral) cores.

This QUP has an SPI/I2C mini cores.

These minicores implements the following logic:

- A common output FIFO provides system output data to one or more minicores
- A common input FIFO provides system input data from one or more minicores

These cores uses BAM (Bus access module) to move data to/from the peripheral buffers. There is a pair of BAM pipes attached to each peripheral.

BAM block periodically reads the input FIFO until the programmed number of words are received.

BAM block periodically populates the output FIFO until the programmed number of words are written to the output FIFO.

Interrupt is generated once transfer completes.

Two H/W Pins of SPI are shared with Aux UART, so SPI and Aux UART couldn't be used at the same time. SPI is disabled by default and S/W change is needed to enable SPI by customer's requirement.

3.2.9. GPIO-Keys

The GPIO-keys module provides a way for an application, running on the Linux side, to listen to interrupts on a GPIO. TGPIO 1-6 can be used for this purpose. GPIO 7-10 are currently configured for UART and hence cannot be used.

Application can then listen to /dev/input/event1 to get the interrupt and the interrupt data.

There are several GPIOs that are able to wake up the system from sleep. When using such a GPIO with the GPIO-KEYS driver, any interrupt on this line will wake the system. Using a GPIO that is not capable of waking up the system with the GPIO-KEYS driver will PREVENT THE SYSTEM FROM GOING INTO SLEEP (the logic is very simple: if there is an interrupt pending on a non-wakeup capable GPIO, do not go to sleep).

The GPIO-Keys module have two parameters:

- tgpios – An array of tgpios to listen on. For example, tgpios=3,5 will have the driver to listen to tgpio3 and tgpio5.
- pull_arr – An optional array of pull settings to apply to each tgpio used. Available options are:
    - 0 – No Pull
    - 1 – Pull Up
    - 2 – Pull Down
    - 3 - Default

For example, to start the gpio-keys driver listen on tgpio 3 (no pull) and tgpio 5 (pull up), use the following command: “modprobe gpio-keys tgpios=3,5 pull_arr=0,1”

The number of tgpios parameters must match the number of pull_arr parameters, otherwise pull_arr is totally ignored.

The following GPIOs are wake up capable (All other Tgpios are not wakeup capable):

- Tgpio-1
- Tgpio-4
- Tgpio-5

3.2.10. Serial interfaces

Apps to external MCU

The LE910Cx includes serial interfaces (cdc-acm and raw data interfaces) over both the UART and USB physical interfaces. These interfaces are accessible from the Linux side and can be used to communicate with an external MCU.

The UART device nodes are:

- /dev/ttyHS0 – This is the high speed UART (the modem port). This port is in use by the modem as an AT command port. An application that wants to use this port for communication with an external MCU should first stop the modem service that runs on top of this port using the following command: “/sbin/ds_uart_script stop”
- /dev/ttyHSL0 – This is the debug console port. This port is in use by default as a console (except for performance builds that disables the console capabilities). This port cannot be relocated to othert purposes in a non-perf build.

The USB device nodes are:

- /dev/ttyGSX (X is a number) – Depending on the selected USB composition, there might be one or several ttyGS ports. For example, USB composition 1201 includes a single ttyGS0 port (in 1201 this port is meant for nmea sentences sent by the modem). This is a gadget serial interface and can be used as a standard tty port to communicate with an external MCU if nmea sentences are not enabled.

ttyGS ports represents:

- **Nmea ports** - These ports are used by the modem to dump nmea sentences. This port is disabled by default, and are enabled when the user activates it via the following AT commands:
    - at$gpsp=1
    - at$gpsnmun=1,1,1,1,1,1,1

If nmea ports are not enabled, these ports can be used by the application to communicate with an external MCU.

There is no other device nodes rather than ttyGS for USB. The USB driver does not export any other USB interface via the dev file system, i.e. no ttyUSB ports will be shown for the diag/AT/SAP usb interfaces.

Apps to Modem

Modem and apps are communicating via shared memory. The devices used for such communication are smd devices and are present in the /dev/node. It is not recommended to work with these ports directly, Telit uses these ports for several use cases and a change there might break these functionalities.

The following is the list of smd channels available on the apps:

```
/dev # ls -l /dev/smd*
crw-rw----    1 root     root      248,  11 Jan  1  1970 /dev/smd11
crw-rw----    1 root     root      248,   2 Jan  6 01:58 /dev/smd2
crw-rw----    1 root     root      248,  21 Jan  1  1970 /dev/smd21
crw-rw----    1 root     root      247,   1 Jan  1  1970 /dev/smd22
crw-rw----    1 root     root      248,  36 Jan  1  1970 /dev/smd36
crw-rw----    1 root     root      248,   7 Jan  1  1970 /dev/smd7
crw-rw----    1 root     root      248,   8 Jan  6 01:58 /dev/smd8
crw-rw----    1 root     root      248,   9 Jan  1  1970 /dev/smd9
```

/dev/smd8 is used by the ttyHS0 for AT commands over the UART.

/dev/smd9 is used by the MCM_ATCOP for AT commands sent via the mcm interfaces.

/dev/smd2 is free for customer use in all LE910Cx models (not available for all LE910C1 models). User can send an AT command via this smd channel and capture a response as following:

```
/dev # cat /dev/smd2&

/dev # echo -e "at\r\n" > /dev/smd2

/dev # at

OK


/dev # echo -e "at+cpin?\r\n" > /dev/smd2

/dev # at+cpin?

+CPIN: READY

OK
```

Other smd channels are used internally by Telit/QC and cannot be accessed by user.

3.2.11. Audio

Audio subsystem

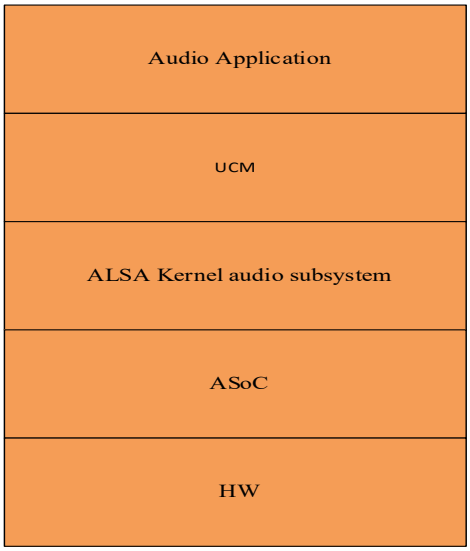
- Optional embedded analog codec with two microphone inputs
- Optional embedded analog codec with one stereo or two mono outputs
- PCM digital audio IO
  - Clock Type: Master
  - Short frame sync
  - Data format 16 –bit linear

  - <clock>
    - 128 – DVI Clock activated at 128KHz
    - 256 – DVI Clock activated at 256KHz
    - 512 – DVI Clock activated at 512KHz
    - 1024 – DVI Clock activated at 1024KHz
    - 2048 – DVI Clock activated at 2048KHz
    - 4096 – DVI Clock activated at 4096KHz
  - <samplerate>
    - 8KHz
    - 16KHz

The Audio user space Application purpose is to:

1. Configure and control the audio properties such as Volume, clock, device mute etc.
2. Use UCM file to configure audio paths and audio devices (handset, hands-free ...)
3. Use AMIX to send commands and data to the ALSA.
4. Manage all voice call events for audio.
5. Manage the digital (external codec) and analog (internal codec) use cases.

Data communication between MAXIM9867 and MDM9628 is via a dedicated Aux PCM interface.



UCM is the user space library that provides C API for clients.

ALSA is the kernel audio subsystem which abstracts the PCM and controls.

ASoC is the low-level driver subsystem.

The ACDB file contains audio calibration data and is categorized as following:

- Handset_cal.acdb − Contains calibration data for handset devices
- Speaker_cal.acdb − Contains calibration data for speakerphone (handheld, handsfree) devices
- Headset_cal.acdb − Contains calibration data for headset devices
- Bluetooth_cal.acdb − Contains BT devices for voice and audio (not supported)
- Global_cal.acdb − Contains voice and audio stream calibration that is independent of device, and other global calibration
- Hdmi_cal.acdb − Contains HDMI audio playback device (not supported)
- General_cal.acdb − Contains general calibration data not falling in any of the other categories.

The LE910Cx has several locations to search ACDB files, which allows customer to override Telit defaults as well as work with QC tools. These ACDB sets of files are searched in the order below:

1. /data/acdb/acdbdata. This is usually where the QC tool put the calibration data when used.
2. Code defaults. This is Telit defaults. We will fall back into this if none of the above exists.

This basically allows the customer to calibrate audio via the QC tool and apply. If the calibration data is accepted, the customer puts the calibration files into its rom volume and the device will automatically pick this up in the next power up.

3.2.11.1. Command-line examples for using the ALSA soundcard driver

- Record an active voice call (analog audio):

- amix 'MultiMedia1 Mixer AUX_PCM_MAX9867_UL_TX' 1
- amix 'Input Mixer' 1
- arec -D hw:0,0 -R 8000 -C 1 file.wav

- Record an active voice call (digital audio):
  - amix 'MultiMedia1 Mixer AUX_PCM_UL_TX' 1
  - arec -D hw:0,0 -R 8000 -C 1 file4.wav

- Play an audio file:
  - amix 'AUX_PCM_MAX9867_RX Audio Mixer MultiMedia1' 1
  - aplay filename.wav

3.2.12. UART

The LE910Cx includes a 4-wire UART with data rates up to 3.75 Mbps.

The UART primarily acts as a modem port.

If the Customer does not need a UART modem port, then can access the TTY device created under the dev file-system. This device can be accessed from user space to read/write data to a peripheral connected to the LE910Cx.

The TTY device can be configured with the STTY utility (included).

To tear down the UART to modem connection, use this command:

/sbin/ds_uart_script stop

After this, the ttyHS0 can be directly accessed (/dev/ttyHS0) from Linux user space for read/write to an external processor connected on the other side.

Following baud rates are supported:

300,600,1200,2400,4800,9600,19200,38400,57600,115200,230400,460800,921600,2000000,2500000,3000000,3500000,3750000

3.2.13. USB Interface

The LE910cX includes a USB2.0 compliant Universal Serial Bus (USB) Transceiver, which operates at USB high-speed (480Mbits/sec). By default, the module is configured as a USB peripheral.

The USB port is typically used for:

- Flashing of firmware and module configuration
- Production testing
- Accessing the Application Processor’s filesystem (debug bridge)
- AT command access (2 modem ports)
- High speed WWAN access to external host
- Diagnostic monitoring and debugging
- NMEA data to an external host CPU

The following standardized device classes can be supported:

- CDC-ACM, CDC-ECM, MBIM, RMNET (Qualcomm proprietary)

The following USB compositions are available:

1200 - NONE

1201 - DIAG + ADB + RMNET + NMEA + MODEM + MODEM + SAP

1203 - RNDIS + DIAG + ADB + NMEA + MODEM + MODEM + SAP

1204 - DIAG + ADB + USB_MBIM + NMEA + MODEM + MODEM + SAP

1205 - USB_MBIM

1206 - DIAG + ADB + ECM + NMEA + MODEM + MODEM + SAP

1250 - RMNET + NMEA + MODEM + MODEM + SAP

1251 - RNDIS + NMEA + MODEM + MODEM + SAP

1252 - USB_MBIM + NMEA + MODEM + MODEM + SAP

1253 - ECM + NMEA + MODEM + MODEM + SAP

1254 - MODEM + MODEM

1255 - NMEA + MODEM + MODEM + SAP

The USB composition can be changed using the usb_composition utility on the Linux side as well as using the #usbcfg AT command (covered by the LE910Cx AT user guide). The usb_composition utility can be used to select any of the above compositions list.

The 125X USB compositions are made specifically for customers that are willing to allow the LE910Cx device to sleep even if the USB is connected (Enables the USB selective suspend). The USB selective suspend also depends on the host implementation.

3.2.14. RTC

LE910cX includes an RTC (Real Time Clock), allowing an application to set an alarm to either wake up the module from sleep mode, power up the module or simply interrupt once the alarm expires. The interface to the RTC, from SW perspective, is made via the RTC device exposed in the device file system (standard RTC device interface).

The RTC driver handles the RTC time and alarm requests from the user.

The RTC sysfs interface can be seen at: /sys/class/rtc/rtc0/

User can set the RTC for wake alarm (using RTC_WKALM_SET ioctl) and receive the notification through the RTC device interface (/dev/rtc0). Another way is to set the RTC wake alarm, shutdown the module (shutdown –h now) and the device will power up automatically when the alarm expires.

In order to allow user space to know why the device has powered up, the following file was added:/var/volatile/tmp/rtc_wake_status.

If the file contains 0 as value, this basically means a normal user selected power up. 1 means power up due to RTC alarm expiration.

3.2.15. Time Services

The time daemon manages the system time between the modem and the apps processor.

The time daemon service starts during boot, and it reads the RTC time from /dev/rtc0 and the offset from file /data/time/ats_1 (or /var/tmp/time/ in older versions). This value is then set as the system time.

When the system boots and time daemon initializes for the first time, the daemon reads the file at /data/time/ats_1 to initialize the offset value. Therefore, as long as the files are present and contain the correct values, the system time is preserved after system reboot.

3.2.15.1. Coordinated Universal Time (UTC)

The UTC time standard is used to regulate the world clock and time. UTC is based on International Atomic Time (TAI), and time zones around the world are expressed as positive or negative offsets from UTC.

UTC is the time standard used for many Internet and web standards. The Network Time Protocol, designed to synchronize the clocks of computers over the Internet, encodes times using the UTC system.

UTC is helpful to avoid confusion about time zones and daylight saving time.

3.2.15.2. Time zones

The time zones around the world are expressed as positive or negative offsets from UTC.

3.2.15.3. Daylight Saving Time (DST)

DST is observed in some time zones by advancing the time by some duration in summer and reverting to the original time during winter. DST helps in getting more daylight in the evening and less in the morning.

3.2.15.4. Time update

Mobile devices have an internal timer system and the time and time zone can be updated manually. Due to inaccuracies of the device-specific internal timer system, each device deviates with time over a period. There are various network standards to synchronize the device time with the network-provided wall clock time periodically:

Network Time Protocol (NTP)

NTP is a networking protocol for synchronizing the clocks of computer systems over packet-switched, variable-latency data networks.

NTP provides UTC, including scheduled leap-second adjustments. No information about time zones or DST is transmitted. This information is outside the scope of this document and must be obtained separately.

NTP uses an epoch of January 1, 1900.

Network Identity and Time Zone (NITZ)

NITZ is a mechanism for provisioning the local time, date, and network provider identity information to mobile devices via a wireless network. NITZ is currently an optional part of the official GSM standard (phase 2+release 96).

The NITZ standard allows the network to “transfer its current identity, universal time, DST, and LTZ, but each is optional and support varies across radio access network vendor and operators.”

With NITZ, the accuracy of the time information is in the order of minutes.

GPS – Clock synchronization

The GPS receiver can also use time information received from GPS time signals. The time accuracy of GPS network is < 1 ms.

3.2.15.5. System time base

Real-time time base

The real time is a wall clock time used by the UI to display the current system time. Generally, this time is updated and synced with time information from the network, e.g., as the modem uses standard network time protocols, such as NTP or NITZ. During sync, this time can go forward or backward with regards to the network time.

Monotonic time base

The monotonic time is time elapsed since epoch time (01 Jan 1980) or some random start time. As this time is not used for UI display or alarm setup, the start time is not relevant.

The monotonic time always moves forward in time but does not reflect wall clock time in any specific way. In the current implementation, CLOCK_MONOTONIC resembles the jiffies tick count in that it starts at 0 when the system boots and increases monotonically from there.

The monotonic time is used by the kernel and user for all relative time events, because this time never goes backward.

3.2.16. Data connection

3.2.16.1. QCMAP_CLI

QCMAP_CLI is a QC user space application allowing to activate a PDP context along with network interface setup, firewall configuration, DNS etc.

This is a GUI application which resides in the /usr/bin directory.

3.2.17. WatchDog

A watchdog is a fixed-length counter that enables a system to recover from an unexpected hardware or software catastrophe.

Unless the system periodically resets the watchdog timer, the watchdog timer assumes a catastrophe and tries to handle the situation.

In general, there are two kinds of watchdog implementations, an hardware watchdog and a software watchdog based on timer interrupt.

Both software and hardware watchdogs are used in the system.

The software watchdog is not implemented in all of the subsystems; e.g., RPM, TrustZone, and APSS do not have software watchdogs, while CNSS and MPSS implement software watchdog.

The hardware watchdog module is a piece of hardware that is used to ensure that the processor is not stuck or overloaded, and consists of a timer that counts down from a predetermined value.

If the timer is not reset (also known as a dog_kick or petting the dog or servicing the dog) by the corresponding CPU core, it eventually counts to 0 and triggers a watchdog timeout.

It is the responsibility of each CPU core to ensure that it keeps resetting the counter. If it is unable to do so (if the dedicated task is starved or the CPU core is locked up, etc.), it is assumed that the system has gone into a bad state.

In addition to the reset triggering signal (WatchDog_expired), it is also possible to generate a watchdog interrupt before the watchdog expiration to allow a processor to attempt the recovery of the system before resetting it:

- BARK (FIQ) – Interrupt before watchdog expiration to allow processor to attempt the recovery of the system before resetting it
- BITE (Reset) – When watchdog timeout happens

The watchdog timer continues counting even after BARK occurs.

3.2.17.1. Hardware watchdog characteristics

In the LE910cX platform, the system has one and only one hardware watchdog counter for every processor or processor cluster located in the corresponding processor subsystem.

- The sole exception is that the apps processor has a non-secure and secure watchdog.
- Every processor or processor cluster is responsible for periodically resetting (kicking/ petting/servicing) its own watchdog.
- The system has one and only one secure watchdog whose primary event (Bite) causes a system reset or cold boot.
- With the exception of the secure watchdog, the primary event, or bite, shall be sent to the apps processor as an interrupt.
- The secondary event, or bark, must and will be serviced by the corresponding processor.
- The secure watchdog is always under the control of the Secure Root of Trust (SRoT).
- The system provides a secure watchdog enable pin to be used for gating the watchdog bites going to the global reset circuitry for debug purposes.
- A watchdog enable pin is provided behind a GPIO.
- The GPIO pin is sensed by the fuse sense operation, and the value is stored in a register in the security control block.
- A watchdog enable fuse is also provided that will be ORed with the GPIO. Hence, if the watchdog enable fuse is blown, the secure watchdog is forced to be enabled and the GPIO is essentially ignored.
- The watchdog timer never causes a processor from exiting a sleep or power-collapse state with the exception of secure watchdog.
- With the exception of programming the watchdog duration, the watchdog hardware is designed such that corresponding software should only need to pet its own watchdog. All of the enabling and halting of the watchdog due to power collapse, processor sleep, and debug conditions is handled automatically by the hardware.
- All of the hardware watchdogs operate on the 32 kHz sleep XTAL clock.
- The size of the watchdog timer counter is 20 bits wide. With sleep clock running at 32 kHz, this gives maximum duration of 32 sec.
- When a processor is in the Reset state, the associated watchdog is held in the Disabled state. When a processor is hung, the subsequent restart process resets the processor subsystem, not just the CPU; hence the watchdog timer including the configuration registers is also reset.
- At cold boot or subsystem restart, the watchdog is held in the Disabled state until the software enables the watchdog timer.

3.2.17.2. Watchdog connectivity

All processor subsystems have the watchdog bite sent to the apps processor. The bark event is routed as a local interrupt.

Note that the non-secure watchdog bark in the apps processor is sent to HLOS, and the non-secure bite is sent to TrustZone.

3.2.17.3. A7 apps processor watchdogs

The A7 apps processor subsystem has two watchdogs (secure and non-secure), each handled by a different software block.

Non-secure A7 apps processor watchdog

If the A7 apps processor hardware watchdog goes without being petted for 4 sec, a bark IRQ is asserted. This is known as nonsecure dog bark.

The kernel handles this as a software error. If the watchdog goes for 5 sec without being petted or the bark IRQ was not handled properly, FIQ is asserted to TrustZone. This is known as nonsecure dog bite and it collects register values, flushes cache contents, and resets the device. The watchdog driver currently schedules itself to pet the watchdog every 3 sec.

For debugging purposes, you can disable the watchdog module by the following method:

- If the watchdog is enabled at bootup (enable=1), the runtime_disable sysfs node at /sys/devices/b017000.qcom,wdt/disable can be used to deactivate the watchdog

Note the dog bark is only received if the processor is not stuck with interrupts enabled and the hardware is still in a functional state, which is not common. If the dog bark cannot be handled and the system remains wedged for whatever reason, the dog bite occurs and resets the target. Register information stored by TrustZone is used to debug non-secure apps dog bite issues.

Secure watchdog timer

The secure watchdog timer is managed by SRoT running on the apps processor.

The bark event of this timer is sent to SRoT as an interrupt. If SRoT ever becomes nonresponsive, the integrity and stability of the entire SoC is in question and the only course of action is a reset to cold boot. Therefore, the bite event of this timer resets the PS_HOLD register, causing the PS_HOLD to transit to a logic-0 state.

This causes the PMIC to enter a chipset reset sequence.

The registers in the secure watchdog timer shall only be accessible by SRoT.

The access control infrastructure must prevent any other EEs than SRoT from accessing the secure watchdog.

The secure watchdog timer shall resume automatically by hardware upon any wakeup event to SRoT when SRoT is in Sleep mode.

The secure watchdog timer shall be held in the Disabled state when the chip is coming out of cold reset until SRoT enables it. **Since only the secure watchdog timer can cause the PMIC to reset the MSM**, it is also used to respond to failures to exit chip sleep states (XO shutdown and/or VDD minimization) by clearing PS_HOLD to cause PMIC to reset the MSM. The secure watchdog timer shall resume automatically on any wakeup event, which directs the chip to exit a

chip sleep state (VDDmin or XO shutdown). The bark of the secure watchdog is an interrupt to SRoT only.

3.2.17.4. MPSS watchdog

The MPSS watchdog module can generate BARK and BITE events:

- BARK (FIQ) – Bark time is programmable
- BITE (Reset) – Watchdog BITE generates an IRQ for the apps processor; MPSS watchdog BITE time is fixed at **~390 ms**

An FIQ is generated on hitting BARK time. A BARK event is handled as a software error in the MPSS. Therefore, the MPSS follows generic error handling procedures.

When the MPSS watchdog timer count reaches 0, the watchdog hardware asserts an interrupt to the apps processor (A7 apps processor). An NMI is generated locally. As the name NMI suggests, this interrupt cannot be blocked. The NMI handler on the subsystem then triggers the error handler that saves the context of the system and informs the apps processor of the error.

BITE interrupts on the MPSS indicate unresponsive subsystems due to a software glitch and/or hardware error such as bus lockup, memory corruption, etc. The A7 apps processor generates a kernel panic once it receives a BITE interrupt from MPSS unless the subsystem restart feature is enabled. In addition to hardware watchdog support, the modem implements the software watchdog, which consists of the DOG task, and a protocol imposed on the reporting tasks in the system. The monitored tasks call a function in order to report to DOG. The DOG software task is one of the highest tasks in the system. It defines a software timer, e.g., 100 ms, which causes it to run periodically to reset the hardware watchdog timer, i.e., pet the hardware watchdog, and prevent the hardware watchdog timeout. Each monitored task has a defined timeout period during which it must report to the DOG task in order to indicate proper operation. These tasks generally set a software timer to schedule their reporting to the DOG task. Since task latency delays make task scheduling and execution uncertain, the tasks must report significantly earlier than their timeout period to reliably avoid a dog timeout.

Periodically, e.g., once every 100 milliseconds, the DOG task checks on the reporting status of the monitored tasks in order to detect overdue task reporting (dog timeout).

3.2.18. Power Management

Hardware Power Architecture/Features

- Single XO
  - **CXO** (19.2 MHz) sources all clocks. CXO is always ON as its buffers are turned off when XO shutdown is exercised. The sleep clock (32 KHz) used by the MDM chipset in XO Shutdown mode is derived from CXO; it is used to clock always-on domains such as the MDM Power Manager (MPM), parts of the modem core, and certain timer circuits.

- Multiple voltage domains – Memory, digital, and APC
  - **VddMx** (memory rail) – VDD_MX represents on-chip memories. This power domain is never switched off but can be minimized to a lower power, typically when MDM is in deep sleep with XOs turned off (memory voltage does not scale in Low-Power modes with the logic domains).

- VddCx (digital rail) – VDD_CX represents all digital logic circuits/cores of MDM that must retain a minimum voltage, even when MDM is in deep sleep (with XOs off). RPM core, system fabric, and MDM9x07 digital cores are some of the examples that depend on this power domain.
- Vdd APC (APC rail) – VDD_APC rail powers up the APSS subsystem.

Software Power Features

- XO shutdown
  - During XO shutdown, CXO buffers at PMIC are switched off, leaving CXO crystal still running.
  - CXO still needs to be generated as it is used to generate the 32 KHz sleep clock in PMIC.
  - No clock is fed to MDM9x07 except the 32 KHz sleep clock, which is required for the always-on parts; for example, MPM block.
  - To enter XO shutdown, all master processors must be in Power collapse mode, dedicated clocks should be off, and shared clocks should have “off” vote from their clients.

- Vdd minimization
  - Vdd minimization is the deepest Low-power mode that can be achieved in the system by minimizing VddCx and VddMx to their lowest possible voltage.
  - When Vdd minimization is achieved, the chip is not operational, except for detecting wake-up interrupt/timer expiration; however, all hardware (supplied by Vdd Cx/Mx) states are still maintained. These two power domains cannot be power-collapsed, so they are put to a lower voltage that can sustain the contents in memories, and so on.
  - Lowering the voltage saves leakage current, and therefore reduces overall power consumption.
  - To enter into Vdd minimization, all master processors should vote for XO Shutdown as well as retention voltage on Cx/Mx.

APPS Power features

Supported Low-Power modes

- **SWFI** – SWFI is an ARM-supported instruction. In this mode, the processor clock is stopped, no instruction is executed, and register states are preserved. Core exits this Low-power mode upon any interrupt.
- **Standalone power collapse** (without RPM notification) – Clock gate and power collapse for individual core. Processor clock is stopped and power-collapse for individual core without assistance from RPM.
- **Power collapse** (with RPM notification) – This Low-power mode includes switching off the power of the core in addition to stopping the processor clock.
  - RPM-assisted power-collapse.
  - Subsystems vote for Low-power mode on shared resources such as XO, L2 cache, and Vdd Cx/Mx to the RPM.
  - There is a higher recovery penalty to resume Active state

Suspend Features

Wakeup Sources Replacing Legacy Android Wakelock

- Enables drivers (kernel space/user space) to finish transactions in progress before going into suspend
- To verify active wakeup sources: cat /sys/kernel/debug/wakeup_sources
- Wakeup sources are good for tracking kernel-originated events, but they do not provide any way for user space to indicate that the system should not sleep
- An application can write a name (and an optional timeout) to /sys/power/wake_lock to establish a new, active wakeup source. The wake lock name provided will be written to the wakeup sources. That source will prevent system suspend until either its timeout expires or the same name is written to/sys/power/wake_unlock.
- A driver receiving a wakeup event will mark the associated wakeup source as active, keeping the system running. That source will stay active until user space has consumed the event. But, before doing so, the user-space application takes a "wake lock" of its own, ensuring that it will be able to complete its processing before the system goes back to sleep.

Autosleep

- Automode is enabled or disabled by writing on/off to /sys/power/autosleep (libsuspend)
- try_to_suspend() function is the core of Autosleep
- Continually looping (without polling) over itself to see if all active wakeup sources have been released while Automode is on

CPUIdle

CPUidle is a kernel PM infrastructure.

CPUs today support multiple idle levels differentiated by varying exit latencies and power consumption during idle.

CPUidle enables efficient management of these different idle CPUs.

It separates out drivers that can provide support for multiple types of Idle states and policy governors that decide what Idle state to use at runtime.

CPUidle driver can support multiple idle states based on parameters such as varying power consumption, wakeup latency, etc.

The main advantage of this infrastructure is that it allows independent development of drivers and governors and allows for better CPU PM.

CPUidle provides idle state detection capability and can also support entry/exit into CPU Idle states

CPUidle initializes cpuidle_device structure for each CPU device and registers with cpuidle using cpuidle_register_device

CPUFreq

CPU Frequency (CPUFreq) scaling is PM method used in running mode.

Sysfs interface – /sys/devices/system/cpu/cpu*/cpufreq/*

Switching of CPUFreq is governed by these governors in static or dynamic manner:

- Performance – CPU runs at static maximum frequency
- Powersave – CPU runs at static minimum frequency
- User space – Determined by static user space program
- Ondemand – On-demand dynamic governor sets target frequency based on CPU busy/idle statistics

- Conservative – Conservative dynamic governor sets target frequency, based on CPU busy/idle statistics

User space interface

LE910Cx provides the following power management capabilities, accessible to the user space:

- Wakelocks. Just like the LE910, a wakelock is used to prevent the system from going into sleep and is accessible to a user space application. Wakelock is implemented as a wakeup source.
- Autosleep. Allows the user to override the above wakelock mechanism.
    - Setting to “off” will force the system not to sleep regardless of the wakelocks.
    - Setting to “mem” will make the system use the wake locks to decide whether to sleep or not.

3.2.19. Performance build

The LE910Cx offers a performance optimized build along with the non-performance optimized build provided so far. Any tests performed on the A4 regarding performance, especially on the following topics, should be performed on the performance build.

The main differences between these two builds are the powerup time and throughput in various scenarios, including WWAN, WLAN and Ethernet.

Performance optimized build changes (oppose to the non perf) the following images:

- Linux LK
- Linux Kernel
- Linux root FS

The following configurations are removed when using the performance build:

- CONFIG_KALLSYMS_ALL - This option makes sure that all symbols are loaded into the kernel image (i.e., symbols from all sections) in cost of increased kernel size
- CONFIG_PROFILING – This option enables the extended profiling support mechanisms used by profilers
- CONFIG_NETFILTER_DEBUG – This option adds additional messages useful in debugging the netfilter code
- CONFIG_DYNAMIC_DEBUG – This option compiles debug level messages into the kernel, which would not otherwise be available at runtime.
- CONFIG_DEBUG_PAGEALLOC – This option Unmaps pages from the kernel linear mapping after free_pages(). This results in a large slowdown, but helps to find certain types of memory corruptions.
- CONFIG_DEBUG_KMEMLEAK – This option enables the memory leak detector and introduces an overhead to the memory allocation.
- CONFIG_DEBUG_KMEMLEAK_DEFAULT_OFF
- CONFIG_DEBUG_STACK_USAGE – This option enables the display of the minimum amount of free stack which each task has ever had available.
- CONFIG_DEBUG_MEMORY_INIT – This option enables additional sanity checks during memory initialization.
- CONFIG_DEBUG_SPINLOCK – This option enables catching missing spinlock initialization and certain other kinds of spinlock errors commonly made.
- CONFIG_DEBUG_MUTEXES – This option allows mutex semantics violations and mutex related deadlocks (lockups) to be detected and reported automatically.

- CONFIG_DEBUG_ATOMIC_SLEEP – When enabling this option, various routines which may sleep will become very noisy if they are called inside atomic sections: when a spinlock is held, inside preempt disabled sections, inside an interrupt, etc.
- CONFIG_DEBUG_LIST – This option enables extended checks in the linked-list walking routines
- CONFIG_FAULT_INJECTION_DEBUG_FS – This option enables configuration of fault-injection capabilities via debugfs.
- CONFIG_FAULT_INJECTION_STACKTRACE_FILTER – This option provides stack trace filter for fault-injection capabilities.
- CONFIG_BLK_DEV_IO_TRACE – This option enables tracing the block layer actions on a given queue. Tracing allows you to see any traffic happening on a block device queue.
- CONFIG_DEBUG_USER - When a user program crashes due to an exception, the kernel can print a brief message explaining what the problem was. This is sometimes helpful for debugging but serves no purpose on a production system.

In addition to the above, the perf build also disabled the debug console starting from the Linux LK through the Kernel image. The console disabling is making almost 90% of the differences in bootup time between these two builds. As a result, in a performance build, there would be no messages printed out to the serial console and no login services would be provided using the serial console.

3.3. Basic Operations

3.3.1. Command Syntax

In the next paragraphs the following notations are used:

<cr> represents the Carriage Return Character (13)

<lf> represents the Line Feed Character (10)

<xx> represents a parameter with changing name is in place of the double x. (< and > characters are only for limiting the parameter and must not be issued to the terminal).

[<xx>] represents an optional parameter whatever name is in place of the xx.

[and] characters are only for limiting the optional parameter and must not be issued to the terminal).

3.3.2. Command Response Timeout

Every command issued to the Telit modules returns a result response if response codes are enabled (default). The time needed to process the given command and return the response varies, depending on the command type. Commands that do not interact with the SIM/UICC or the network, and involve only internal set up settings or readings, have an immediate response, depending on SIM/UICC configuration (e.g., number of contacts stored in the phonebook, number of stored SMS), or on the network the command may interact with.

In the table below are listed only the commands whose interaction with the SIM/UICC or the network could lead to long response timings. When not otherwise specified, timing is referred to set command. For phonebook and SMS writing and reading related commands, timing is referred to commands issued after phonebook sorting is completed. For DTMF sending and dialing commands timing is referred to module registered on network (“AT+CREG/+CEREG?” answer is “+CREG/+CEREG: 0,1” or “+CREG/+CEREG: 0,5”).

In case no response is received after the timeout time has been elapsed, then try repeating the last command and if still no response is received until the timeout time, an Unconditional Shutdown MUST be issued and the device must be powered ON again.

Command	Time-Out (Seconds)
+COPS	125 (test command)
+CLCK	15 (SS operation)
	5 (FDN enabling/disabling)
+CPWD	15 (SS operation)
	5 (PIN modification)
+CLIP	15 (read command)
+CLIR	15 (read command)
+CCFC	15
+CCWA	15
+CHLD	60
+CPIN	30
+CPBS	5 (FDN enabling/disabling)
+CPBR	5 (single reading)
	15 (complete reading of a 500 records full phonebook)
+CPBF	10 (string present in a 500 records full phonebook)
	5 (string not present)
+CPBW	5
+CACM	5
+CAMM	5
+CPUC	180

Command	Time-Out (Seconds)
+VTS	20 (transmission of full “1234567890*#ABCD” string with no delay between tones, default duration)
+CSCA	5 (read and set commands)
+CSAS	5
+CRES	5
+CMGS	120 after CTRL-Z; 1 to get ‘>’ prompt
+CMSS	120 after CTRL-Z; 1 to get ‘>’ prompt
+CMGW	5 after CTRL-Z; 1 to get ‘>’ prompt
+CMGD	5 (single SMS cancellation)
	25 (cancellation of 50 SMS)
+CNMA	120 after CTRL-Z; 1 to get ‘>’ prompt
+CMGR	5
+CMGL	100
+CGACT	150
+CGATT	90
D	120 (voice call)
	Timeout set with ATS7 (data call)
A	60 (voice call)
	Timeout set with ATS7 (data call)
H	60
+CHUP	60
+COPN	10
+COPL	180
+WS46	10
+CRSM	180
#MBN	10
#TONE	5 (if no duration specified)

Command	Time-Out (Seconds)
#EMAILD	90
#STSR	30
#GPRS	150
#SKTD	140 (DNS resolution + timeout set with AT#SKTCT)
#QDNS	170
#FTPOPEN	120 (timeout set with AT#FTPTO, in case no response is received from server)
#SGACT	150
#SH	10
#SD	140 (DNS resolution + connection timeout set with AT#SCFG)
#CSURV	125
#CSURVC	125
#CSURVUC	125
#CSURVB	125
#CSURVBC	125
#CSURVP	125
#CSURVPC	125

3.3.3. Basic AT Commands

3.3.3.1. AT Error Report Format

Disable the Error Report in numerical and verbose format.

AT+CMEE=0

OK

Enable the Error Report in numerical format.

AT+CMEE=1

OK

Enable the Error Report in verbose format.

AT+CMEE=2

3.3.4. RAT and Band Selection

3.3.4.1. RAT selection

The following AT command selects the technology: 2G, 3G, 4G, or automatic.

AT+WS46=[<n>]

<n> - integer type, it is the WDS-Side Stack used by the TA.

12 GSM Digital Cellular Systems (GERAN only)

22 UTRAN only

25 3GPP Systems (GERAN and UTRAN and E-UTRAN)

28 E-UTRAN only

29 GERAN and UTRAN

30 GERAN and E-UTRAN

31 UTRAN and E-UTRAN

The <n> parameter is stored in NVM, and the command will take effect on the next power on.

The factory default value depends on each variant.

3.3.4.2. Band selection

In manual band selection the following AT command selects the current band for both technologies GERAN, UTRAN and EUTRAN:

AT#BND=<GSM band>[,<UMTS band>[,<LTE band>]]

Examples

AT#BND=0,0,2 ➔ selected band: GSM(2G) + DCS(2G) + B1 (3G) + B2(4G)

OK

The input range for supported band are depends on variants.

3.3.5. SIM/USIM Management

3.3.5.1. SIM Presence and PIN Request

The following AT command checks if the SIM device needs the PIN code:

AT+CPIN?

Examples

Assume that the SIM is inserted into the module and the PIN code is needed.

AT+CPIN?

+CPIN: SIM PIN

OK

Assume that the SIM is not inserted and Extended Error result code is not enabled. Check if PIN code is needed, just to see the response command:

AT+CPIN?

ERROR

Assume that the SIM is not inserted and Verbose Extended error result code is enabled. Check if PIN code is needed, just to see the response command:

AT+CPIN?

+CME ERROR: SIM not inserted

Assume that the SIM is not inserted and Numerical Extended error result code is enabled. Check if PIN code is needed, just to see the response command:

AT+CPIN?

+CME ERROR: 10

3.3.5.2. Enter PIN code

Use the following AT command to enter the PIN code:

AT+CPIN=<pin>

Examples

Assume to enter a wrong PIN code, and Extended Error result is not enabled.

AT+CPIN=1235

ERROR

Now, enter the right PIN code:

AT+CPIN=1234

OK

Enable Verbose Extended error result code:

AT+CMEE=2

OK

Enter a wrong PIN code:

AT+CPIN=1235

+CME ERROR: incorrect password.

After 3 PIN code failed attempts, the PIN code is no longer requested and the SIM is locked. Use SIM PUK to enter a new PIN code and unlock the SIM

3.3.5.3. Enter PUK code

Enter the following AT command if PUK or PUK2 code is required:

AT+CPIN=<pin>[,<newpin>]

After 10 PUK code failed attempts, the SIM Card is locked and no longer available.

3.3.5.4. SIM Status

Use the following AT command to enable/disable the SIM Status Unsolicited Indication.

AT#QSS=<mode>

Example 1

Enable the unsolicited indication concerning the SIM status change.

AT#QSS=1 ← enable URCs: #QSS:0/1

OK

#QSS: 0 ← unsolicited indication: SIM is extracted.

#QSS: 1 ← unsolicited indication: SIM is inserted.

Example 2

AT#QSS=2 ← enable URCs: #QSS:0/1/2/3

OK

AT+IPR=19200 ← select the Main Serial Port speed = DTE speed

OK

AT&W0 ← store the setting on profile 0

OK

AT&P0 ← at Power on use profile 0

OK

Now, power off the module:

#QSS: 0 ← unsolicited indication: SIM is extracted.

Now, power on the module:

#QSS: 1 ← unsolicited indication: SIM is inserted.

AT+CPIN?

+CPIN: SIM PIN ← SIM is locked

OK

AT+CPIN=<PIN> ← enter PIN

OK

#QSS: 2 ← unsolicited indication: SIM is unlocked.

#QSS: 3 ← unsolicited indication: SMS and Phonebook are accessible

3.3.5.5. SIM Detection Mode

Use the following AT command to manage the SIM Detection Mode:

AT#SIMDET=<mode>

Or

Use the following AT command to enable/disable the SIM Status Unsolicited Indication.

AT#SIMPR = <mode>

Example

AT#SIMDET?

#SIMDET: 2,1

OK

2 = automatic SIM detection through SIMIN pin (Factory Setting)

1 = SIM inserted

AT#SIMPR?

#SIMPR: 0, 0, 1

#SIMPR: 0, 1, 0

OK

First Line:

0 = Disable URC

0 = Local UICC

1 = SIM inserted

Second Line:

0 = Disable URC

1 = Remote SIM

0 = Remote SIM not connected (If SIM/UICC Access Profile of BT is supported)

Enable the unsolicited indication concerning the SIM status change.

AT#QSS=1

OK

Now, extract the SIM

#QSS: 0 ← unsolicited indication: SIM is extracted

Now, insert the SIM

#QSS: 1 ← unsolicited indication: SIM is inserted

AT#SIMDET=0 ← simulate SIM not inserted, but it is still physically inserted

OK

#QSS: 0 ← unsolicited indication, but SIM is NOT physically extracted

AT#SIMDET?

#SIMDET: 0,1 ← 0 = simulate the status SIM not inserted, 1 = SIM is

OK physically inserted

AT#SIMPR?

#SIMPR: 0, 0, 1 ← 0: Disable URC, 0: Local SIM, 1: SIM inserted

#SIMPR: 0, 1, 0 ← 0: Disable URC, 1: Remote SIM, 0: Remote SIM not inserted

OK

Now, extract/insert the SIM, no unsolicited indication appears on DTE!

Extract the SIM again

AT#SIMDET=1 ← simulate SIM inserted, but it is still physically extracted

OK

AT#SIMDET?

#SIMDET: 1,0 ← 1 = simulate the status SIM inserted, 0 = SIM is physically

OK not inserted

AT#SIMPR?

#SIMPR: 0, 0, 0 ← 0: Disable URC, 0: Local SIM, 0: SIM is physically not inserted

#SIMPR: 0, 1, 0 ← 0: Disable URC, 1: Remote SIM, 1: Remote SIM not inserted

OK

Now, insert/extract the SIM, no unsolicited indication appears on DTE!

AT#SIMPR=1 ← Enable URC

OK

Extract the SIM and set automatic SIM detection

#SIMPR: 0, 0 ← 0: Local SIM, 0: SIM is physically not inserted

#SIMPR: 1, 0 ← 1: Remote SIM, 0: Remote SIM is not connected from SAP

AT#SIMDET=2

OK

AT#SIMDET?

#SIMDET: 2,0 ← 2 = automatic SIM detection through SIMIN pin (Factory Setting),

OK 0 = SIM not inserted

Now, insert/extract the SIM, unsolicited indication appears again on DTE!

#SIMPR: 0, 1 ← 0: Local SIM, 0 SIM is physically inserted

#SIMPR: 1, 0 ← 1: Remote SIM, 0: Remote SIM is not connected from SAP

#QSS: 1 ← unsolicited indication: SIM is logically activated

#QSS: 0 ← unsolicited indication: SIM is logically deactivated

3.3.5.6. SIM/USIM Access File

AT+CSIM command is used to read/write SIM/USIM files. The format of the AT+CSIM parameters and the sequence of the AT+CSIM commands must be in accordance with the protocol card. The distinction between SIM and USIM <command> format is needed because the AT+CSIM command works directly on the card.

AT+CSIM=<length>,<command>

To read/write card files refer to “LE910Cx_AT_Command_Reference_Guide.doc”.

3.3.5.7. MSISDN

MSISDN is a number uniquely identifying a subscription in a GSM or UMTS mobile network. MSISDN is defined by the ITU-U Recommendation which defines the numbering plan: a number uniquely identifies a public network termination point and typically consists of three fields, CC (Country Code), NDC (National Destination Code), and SN (Subscriber Number), up to 15 digits in total.

Select the "ON" storage:

AT+CPBS="ON"

OK

Write a new record on the selected storage:

AT+CPBW=1,"+393912457",145,"MyNumber"

OK

Read the just entered number:

AT+CPBF="MyNumber"

+CPBF: 1," +393912457",145,"MyNumber"

OK

3.4. Advanced Operations

3.4.1. SMS Management

The modules provide the SMS Service to store, send, receive, and delete a SMS, which is a short text message up to 160 characters long. Before using the SMS messages, you must configure the Short Message Service.

3.4.1.1. Select SMS Format Type

The Telit Module supports two SMS formats:

- PDU mode
- Text mode

The module uses the PDU format to send a message on the air. The PDU mode enables the user to edit the message in PDU format. If the user is familiar with PDU encoding, he can operate with PDU by selecting that mode and use the appropriate commands.

The present document uses the Text mode to explain how to operate with SMS. Here is the AT command to select the mode.

AT+CMGF=<mode>

Examples

Check the supported range of values:

AT+CMGF=?

+CMGF: (0,1)

OK

Set up Text Mode for the SMS:

AT+CMGF=1

OK

This setting is stored and remains active until the module is turned OFF.

3.4.1.1.1. Set Text Mode Parameters

When SMS format is Text mode, the SMS parameters that usually reside on the header of the PDU must be set apart with the +CSMP command.

AT+CSMP=<fo>,<vp>,<pid>,<dcs>

Example 1

Set the SMS parameters as follow:

- <fo> expressed in binary format, see table below. The binary number expressed in decimal format is 17.

0	0	0	1	0	0	0	1
Module is not requesting a status report	Always 0	Replay Path not requested	Validity period field present in relative format		Always 0	SMS-SUBMIT	

- <vp> validity period (in relative format) = 24 hours is coded into 167 decimal format.
- <pid> protocol identifier.
- <dcs> data coding scheme, default value 0.

AT+CSMP= 17,167,0,0

OK

Example 2

Set the SMS parameters as follow:

- <fo> expressed in binary format, see table below. The binary number expressed in decimal format is 25.

0	0	0	1	1	0	0	1
Module is not requesting a status report	Always 0	Replay Path not requested	Validity period field present in absolute format		Always 0	SMS-SUBMIT	

- <vp> validity period in absolute format represents the expiration date of the message, for example:

date: 29/06/02; time: 02:20; in the time zone of Italy (+1) is formatted as follows: “29/06/02,02:20:00+1”

- <pid> protocol identifier.
- <dcs> data coding scheme:
  - Default Alphabet
  - Class 0 (immediate display SMS)

Data coding scheme is coded in the following binary format: 11110000, corresponding to 240 in decimal format.

AT+CSMP=25,"29/06/02,02:20:00+1",0,240

OK

Use dcs=0 if no particular data coding scheme is needed. Not all dcs combinations described in the 3GPP TS 23.038 are jointly supported by Networks and Telit Modules: some features may be not implemented on Networks or on Telit Modules. This no matching is resulting in a ERROR result code, use different dcs.

3.4.1.1.2. Character Sets

Use the following AT command to select the character set:

AT+CSCS=<chset>

Here are the supported character sets:

- “GSM” default alphabet
- “IRA” – ITU-T.50
- ”8859-1” – ISO 8859 Latin 1
- ”PCCP437” – PC character set Code Page 437.
- “UCS2” – 16-bit universal multiple-octet coded character set (ISO/IEC10646)

Examples

Check the supported character sets:

AT+CSCS=?

+CSCS: (“GSM”,”IRA”,”8859-1”,”PCCP437”,”UCS2”)

OK

Check the current character set:

AT+CSCS?

+CSCS: “IRA”

OK

Select a non-existent character set, merely to see the response format:

AT+CSCS=”GSA”

ERROR

Enabling the Error report in verbose format:

AT+CMEE=2

OK

Select again a non-existent character set:

AT+CSCS=”GSA”

+CME ERROR: operation not supported

3.4.1.1.2.1. IRA Character Set

The IRA character set is used in Text mode. IRA set defines each character as a 7-bit value: from 0x00 to 0x7F. The table below lists all the supported characters and their hexadecimal code.

		Most Significant Nibble							
		0x	**1x**	**2x**	**3x**	**4x**	**5x**	**6x**	**7x**
Least Significant Nibble	**x0**			SP[1]	0	@	P		p
	x1			!	1	A	Q	a	q
	x2			“	2	B	R	b	r
	x3			#	3	C	S	c	s
	x4			$	4	D	T	d	t
	x5			%	5	E	U	e	u
	x6			&	6	F	V	f	v
	x7			‘	7	G	W	g	w
	x8			(	8	H	X	h	x
	x9			)	9	I	Y	i	y
	xA	LF[2]		*	:	J	Z	j	z
	xB			+	;	K		k	
	xC			,	<	L		l	
	xD	CR[3]		-	=	M		m	
	xE			.	>	N		n	
	xF			/	?	O	£	o	

1 – SP stands for space character

2 – LF stands for Line Feed character

3 – CR stands for Carriage Return character

The following examples show how to use the IRA table:

- Get the IRA code of the character ‘&’: the most significant nibble is 2, the least significant nibble is 6, so the IRA code for the ‘&’ character is the hexadecimal value: 0x26.
- Translate IRA code 0x6B into the corresponding character: the most significant nibble is 6, the least significant nibble is B, the cell at the crossing of column 6 and row B holds the character: "k".

3.4.1.1.2.2. UCS2 Character Set

The UCS2 Character Set is used in Text mode.

- Phone number 329 05 69 6... converted into "UCS2" format: 3=0033, 2=0032, 9=0039, 0=0030, 5=0035, 6=0036, 9=0039, 6=0036 ...

- Text HELLO converted into UCS2 format: H=0048, E=0045, L=004C, O=004F

3.4.1.2. Read SMSC Number

The module sends the SMS to the SMSC where the message is dispatched towards its final destination or is kept until the delivery is possible. To ensure the correct operation of this service the number of the SMSC needs to be configured on the module in accordance with the network operator used.

To know the SMSC number stored on the module, use the following AT command.

AT+CSCA?

Check the stored SMSC number:

AT+CSCA?

+CSCA: "+39X20XX58XX0",145

OK

SMSC number is compliant with the international numbering scheme.

3.4.1.3. Set SMSC Number

Use the following AT command to store a new SMSC number. The old number is overwritten.

AT+CSCA=<number>,<type>

Set up the desired SMSC number in international format:

AT+CSCA=+39X20XX58XX0,145

OK

Enable extended result code in verbose format:

AT+CMEE=2

OK

Enter the command with no parameters:

AT+CSCA=

+CME ERROR: operation not supported

3.4.1.4. Send a SMS

Use the following AT command to send a SMS.

AT+CMGS

To read and set the SMSC number see § 3.4.1.2 and 3.4.1.3.

Example 1

Send a SMS to the module itself and do not store it. Use the UCS2 character set.

Select Text Mode.

AT+CMGF=1

OK

Select the UCS2 character set.

AT+CSCS="UCS2"

OK

Set SMS parameters:

AT+CSMP=17,168,0,26

OK

Select how the new received message event is notified by the DCE to the DTE.

AT+CNMI=1,1,0,0,0

OK

Send the message to the module itself. The UCS2 character set is used:

- Phone number 329 05 69 628 is converted into "UCS2" format: 3=0033, 2=0032, 9=0039, 0=0030, 5=0035, 6=0036, 9=0039, 6=0036, 2=0032, 8=0038
- Text CIAO is converted into UCS2 format: C=0043, I=0049, A=0041, O=004F

AT+CMGS=00330032003900300003500360039003600320038

> 004300490041004F (close the message with Ctrl Z)

+CMGS: 81

OK

The module itself receives the SMS, the following unsolicited indication is shown on DTE:

+CMTI: "SM",3

The SMS was successfully sent to the SMSC and its network reference number is 81. Do not confuse message reference with message index position: the first one indicates the network reference for identifying the sent message, the second one – reported by the unsolicited indication – indicates that the module receives the message and it is stored on the position 3 of the "SM" storage.

Select the "SM" storage as indicated by the unsolicited indication.

AT+CPMS="SM"

+CPMS: 3,50,3,50,3,50

OK

Read the message from the storage position indicated by the unsolicited indication.

AT+CMGR=3

+CMGR: "REC UNREAD","002B00330039003300320039003000350036003900360032003 8",

"00570049004E00440020005300490044D","08/05/13,12:22:08+08"

004300490041004F

OK

Example 2

Send a SMS to the module itself and do not store it.

Select Text Mode

AT+CMGF=1

OK

Select how the new received message event is notified by the DCE to the DTE.

AT+CNMI=1,1,0,0,0

OK

Send the message to the module itself.

AT+CMGS="+39329X569YYY"

> SEND THE SMS #1 TO ITSELF (close the message with Ctrl Z)

+CMGS: 76

OK

The module itself receives the SMS #1, the following unsolicited indication is shown on DTE:

+CMTI: "SM",1

The SMS was successfully sent to the SMSC and its network reference number is 76. Do not confuse message reference with message index position: the first one indicates the network reference for identifying the sent message, the second one – reported by the unsolicited indication – indicates that the module has received the message and it is stored on the position 1 of the "SM" storage.

Use unsolicited indication parameter to read the SMS #1 for the first time.

AT+CMGR=1

+CMGR: "REC UNREAD","+39329X569YYY","WIND SIM","08/04/18,13:58:04+08"

SEND THE SMS #1 TO THE MODULE ITSELF

OK

3.4.1.5. Select/Check SMS Storage Type

Telit Modules can provide two type of SMS storage, in agreement with the family of belonging:

- “SM” – SIM Card Memory
- “ME” – Mobile Equipment Memory

- “SR” – Status Report Message Memory.

Use the following AT command to select memory storage:

AT+CPMS=<memr>,<memw>,<mems>

The SMS are usually stored (this is true for both the originated and the received SMS) in the SM/ME storage.

The LE910Cx family allows the user to select a different storage for the read-delete, write-send, and reception-saving SMS operations.

Examples

AT+CPMS=? ← Check the supported SMS storage types

+CPMS: (“ME”,“SM”,“SR”),(“SM”,“ME”),(“SM”,“ME”)

OK

AT+CPMS? ← Check the current active storage type

+CPMS: “SM”,1,50,”SM”,1,50,”SM”,1,50

OK

AT+CPMS=”ME” ← Select “ME” storage type

+CPMS: 0,50,1,50,1,50

OK

AT+CPMS? ← Check the current active storage types

+CPMS: “ME”,0,50,”SM”,1,50,”SM”,1,50 ←Two SMS storage types are active: “ME” and “SM”

OK

3.4.1.6. Store a SMS

Use the following AT command to store a SMS.

AT+CMGW="<da>"

Example

Store a SMS in the "SM" storage, send it to the module itself and read the message in the receiving storage.

AT+CMGF=1 ← Select Text Mode

OK

AT+CSMP=17,168,0,240 ← Assume to send a SMS of Class 0

OK

Select how the new received message event is notified by the DCE to the DTE

AT+CNMI=1,1,0,0,0

OK

Store into "SM" the SMS message to be sent to the module itself.

AT+CMGW="+39329X569YYY"

> SEND THE STORED SMS #1 TO THE MODULE ITSELF (close with Ctrl Z or ESC to abort)

+CMGW: 5

OK

Use index 5 to read SMS #1 from "SM" storage type.

AT+CMGR=5

+CMGR: "STO SENT","+39329X569YYY","WIND SIM"

SEND THE STORED SMS # 1 TO MODULE ITSELF

OK

Send the stored SMS #1using the storage position 5 returned by the previous command.

AT+CMSS=5

+CMSS: 78

OK

The module itself receives the SMS #1, the following unsolicited indication is shown on DTE:

+CMTI: "SM",6

Check the current SMS storage type.

AT+CPMS?

+CPMS: "SM",6,30,"SM",6,30,"SM",6,30

OK

Use index 6 to read received SMS #1 from "SM" storage memory.

AT+CMGR=6

+CMGR: "REC UNREAD","+39329X569YYY","WIND SIM","08/04/21,09:56:38+08"

SEND THE STORED SMS # 1 TO THE MODULE ITSELF

OK

Use index 6 to read again received SMS #1 from "SM" storage memory.

AT+CMGR=6

+CMGR: "REC READ","+39329X569YYY","WIND SIM","08/04/21,09:56:38+08"

SEND THE STORED SMS # 1 TO THE MODULE ITSELF

OK

3.4.1.7. Send a Stored SMS

A SMS stored in the "SM" storage type is sent using the following AT command. Its storage location index is needed.

AT+CMSS=<index>

Example

Send the stored SMS to the module itself:

Select Text Mode

AT+CMGF=1

OK

Select “SM” storage to read SMS

AT+CPMS=”SM”

+CPMS: 1,50,1,50,1,50

OK

Read the SMS stored on position 1.

AT+CMGR=1

+CMGR: “STO SENT”,”+39329X569YYY”,”WIND SIM”

SEND THE STORED SMS # 1 TO MODULE ITSELF

OK

Select how the new received message event is indicated by the DCE to the DTE.

AT+CNMI=1,1,0,0,0

OK

Send the stored SMS # 1 message to module itself.

AT+CMSS=1

+CMSS: 79

OK

The module itself receives the SMS #1, the following unsolicited indication is shown on DTE:

+CMTI: “SM”,2

3.4.1.8. Delete an SMS

Use the following AT command to delete an SMS stored on the “SM” storage type.

AT+CMGD=<index>

Example

Deleting an SMS stored in “SM” storage type:

AT+CPMS=”SM” ← Select memory storage

+CPMS: 13,50,13,50,13,50

OK

AT+CMGD=? ← Check the SMS

+CMGD: (1,2,3,4,5,6,7,8,9,10,11,12,13),(0-4)

OK

Delete SMS in memory position 1.

AT+CMGD=1

OK

Check if the SMS is deleted:

AT+CMGD=?

+CMGD: (2,3,4,5,6,7,8,9,10,11,12,13),(0-4)

OK

Delete all SMS. Disregard the first parameter of the +CMGD.

AT+CMGD=1,4

OK

AT+CMGD=?

+CMGD: (),(0-4)

OK

3.4.1.9. Read an SMS

An SMS is read with the following command:

AT+CMGR=<index>

Example

AT+CPMS?

+CPMS: “SM”,1,50,”SM”,1,50,”SM”,1,50

OK

Read the SMS #1, for the first time, in storage memory “SM”, position 1:

AT+CMGR=1

+CMGR: “STO SENT”,”+39329X569YYY”,”WIND SIM”

SEND THE STORED SMS # 1 TO MODULE ITSELT

OK

3.4.1.10. SMS Status

SMSs can be gathered into 5 different groups depending on their Status:

- REC UNREAD: received messages still not read
- REC READ: received messages already read
- STO UNSENT: written messages not yet sent
- STO SENT: written messages already sent
- ALL: all types of messages

Use the following AT command to query the SMS status:

AT+CMGL=<stat>

Check if Text Mode is active

AT+CMGF?

+CMGF: 1 ← Text Mode is active

OK

Check the supported SMS status

AT+CMGL=?

+CMGL: (“REC UNREAD”,”REC READ”,”STO UNSENT”,”STO SENT”,”ALL”)

OK

Check the available SMS storage type

AT+CPMS?

+CPMS: “SM”,6,30,”SM”,6,30,”SM”,6,30

OK

List all the SMSs stored on “SM” storage with their Status.

AT+CMGL=”ALL”

+CMGL: 1,”REC READ”, •••• SMS body ••••

+CMGL: 2,”REC READ”, •••• SMS body ••••

+CMGL: 3,”REC READ”, •••• SMS body ••••

+CMGL: 4,”STO SENT”, •••• SMS body ••••

+CMGL: 5,”STO SENT”, •••• SMS body ••••

+CMGL: 6,”REC READ”, •••• SMS body ••••

OK

List the SMSs stored on “SM” storage with their Status=STO SENT

AT+CMGL=”STO SENT”

+CMGL: 4,”STO SENT”, •••• SMS body ••••

+CMGL: 5,”STO SENT”, •••• SMS body ••••

OK

3.4.1.11. Cell Broadcast Service

GSM Standard specifies two different types of SMS:

- SMS Point to Point (SMS/PP),
- SMS Cell Broadcast (SMS/CB).

The first type can send a text message long up to 160 characters from a module to the another (as stated on the previous paragraphs), the second type allows the Network to send, at the same time, a message to all modules contained in the defined area including one or more radio cells. The availability and the implementation of the Cell Broadcast Service are strictly connected with the Network Operator of the subscriber.

Use the following AT command to enable the Cell Broadcast Service:

AT+CSCB=[<mode>[,<mids>[,<dcss>]]]

Select Text Mode.

AT+CMGF=1

OK

Select the District service.

AT+CSCB=0,50,0

OK

Select how the new received message event is indicated by the DCE to the DTE.

AT+CNMI=2,0,2,0,0

OK

After a while the “District” broadcast message is displayed on the DTE.

+CBM: 24,50,1,1,1

TRIESTE

+CBM: 4120,50,2,1,1

TRIESTE

+CBM: 8216,50,1,1,1

TRIESTE

+CBM: 12312,50,2,1,1

TRIESTE

<mids>	Service name
000	Index
010	Flashes
020	Hospitals
022	Doctors
024	Pharmacy
030	Long Distant Road Reports
032	Local Road Reports
034	Taxis
040	Weather
050	District
052	Network Information
054	Operator Services
056	Directory Inquiries (national)
057	Directory Inquiries (international)
058	Customer Care (national)

059	Customer Care (international)

3.4.2. GNSS Management

3.4.2.1. Introduction

The LE910Cx module is equipped with IZat™ Gen 8 that is controllable by the modem using a set of AT commands or dedicated NMEA sentences.

3.4.2.2. LE910Cx Serial Ports

3 serial ports are available on the module:

- MODEM #1 USB SERIAL PORT
- MODEM #2 USB SERIAL PORT
- NMEA USB SERIAL PORT

3.4.2.3. WGS84

The GPS receivers perform initial position and velocity calculations using an earth-centered earth-fixed (ECEF) coordinate system. Results may be converted to an earth model (geoid) defined by the selected datum. For LE910Cx, the default datum is WGS 84 (World Geodetic System 1984) which provides a worldwide common grid system that may be translated into local coordinate systems or map dates. (Local map dates are a best fit to the local shape of the earth and not valid worldwide)

3.4.2.4. NMEA 0183

The NMEA 0183 is a specification created by the National Marine Electronics Association (NMEA) that defines the interface between other marine electronic equipment. The standard permits marine electronics to send information to computers and to other marine equipment. GPS receiver communication is defined within this specification. The actually supported version is **4.10.**

The provided NMEA sentences are:

GGA GPS Fix Data. Time, position and fix type data.

GLL Geographic Position - Latitude/Longitude

GSA GPS receiver operating mode, satellites used in the position solution and DOP values.

GSV The number of GPS satellites in view satellite ID numbers, elevation, azimuth, and SNR values.

RMC Time, date, position, course and speed data.

VTG Course and speed information relative to the ground

GNS GNSS fix data.

GNS Range residuals.

The NMEA USB port provides the following sentences with $GPSNMUN command: GGA, GLL, GSA, GSV, RMC, VTG.

The NMEA USB port provides the following sentences with $GPSNMUNEX command: GNS.

3.4.2.4.1. GGA – Global Position System Fixed Data

This sentence provides time, position, and fixes related data for a GPS Receiver. Table A contains the values for the following example:

$GPGGA,161229.48,3723.247522,N,12158.341622,W,1,07,1.0,72.1,M,18.0,M,,*18

Table A: GGA Data Format

Name	Example	Units	Description
Message ID	$GPGGA		GGA protocol header GP: GPS Talker ID
UTC Time	161229.48		hhmmss.ss
Latitude	3723.247522		ddmm.mmmmmm
N/S Indicator	N		N=north or S=south
Longitude	12158.341622		dddmm.mmmmmm
E/W Indicator	W		E=east or W=west
Position Fix Indicator	1		See Table B
Satellites Used	07		Range 0 to 12
HDOP	1.0		Horizontal Dilution of Precision
MSL Altitude	72.1	meters	Antenna Altitude above/below mean-sea-level (geoid).
Units	M	meters	Units of antenna altitude
Geoid Separation	18.0	meters	The difference between the WGS-84 earth ellipsoid and the mean-sea-level (geoid), "-" means mean-sea-level below ellipsoid.
Units	M	meters	Units of geoidal separation

Age of Diff. Corr.		second	Null fields when DGPS is not used
Diff. Ref. Station ID			Null fields when DGPS is not used / 0000-1023
Checksum	*18		
<CR> <LF>			End of message termination

Table B : Position Fix Indicator

Value	Description
0	Fix not available or invalid
1	GPS SPS Mode, fix valid
2	Differential GPS, SPS Mode, fix valid
3	GPS PPS Mode, fix valid
4	Real Time Kinematic
5	Float RTK
6	Estimated (dead reckoning) Mode
7	Manual Input Mode
8	Simulator Mode

3.4.2.4.2. GLL - Geographic Position - Latitude/Longitude

This sentence provides latitude and longitude of vessel position, time of position fix and status. Table C contains the values for the following example:

$GPGLL,3723.247522,N,12158.341622,W,161229.48,A,A*41

Table C: GLL Data Format

Name	Example	Units	Description
Message ID	$GPGLL		GLL protocol header GP: GPS Talker ID
Latitude	3723.247522		ddmm.mmmmmm

N/S Indicator	N		N=north or S=south
Longitude	12158.341622		dddmm.mmmmmm
E/W Indicator	W		E=east or W=west
UTC Time	161229.48		hhmmss.ss
Status	A		A=data valid or V=data not valid
Mode Indicator	A		See Table D
Checksum	*41		
<CR> <LF>			End of message termination

Table D : Mode Indicator

Value	Description
N	Fix not available or invalid
A	GPS SPS Mode, fix valid
D	Differential GPS, SPS Mode, fix valid
P	GPS PPS Mode, fix valid
R	Real Time Kinematic
F	Float RTK
E	Estimated (dead reckoning) Mode
M	Manual Input Mode
S	Simulator Mode

3.4.2.4.3. GSA - GNSS DOP and Active Satellites

This sentence reports the GPS receiver's operating mode, satellites used in the navigation solution reported by the GGA sentence and DOP values. Table D contains the values for the following example:

$GPGSA,A,3,07,02,26,27,09,04,15, , , , , ,1.8,1.0,1.5,1*33

Table E: GSA Data Format

Name	Example	Units	Description
Message ID	$GPGSA		GSA protocol header

			GP: GPS Talker ID GN: GNSS Talker ID BD: Beidou Talker ID
Mode 1	A		See Table F
Mode 2	3		See Table G
Satellite Used1. Satellite used in solution.1	07		Sv on Channel 1 GPS: 1-32 SBAS: 33-64 (offset 87) GLONASS: 65-96. GALILEO:1-36 (offset 300) BEIDOU:1-37 (offset 200)
Satellite Used1	02		Sv on Channel 2
....			
Satellite Used1			
PDOP	1.8		Position Dilution Of Precision
HDOP	1.0		Horizontal Dilution Of Precision.
VDOP	1.5		Vertical Dilution Of Precision.
GNSS System ID	1		1=GPS 2=GLONASS 3=GALILEO 4=BEIDOU
Checksum	*33		
<CR> <LF>			End of message termination

Table F: Mode 1

Value	Description
M	Manual—forced to operate in 2D or 3D mode
A	2D Automatic—allowed to automatically switch 2D/3D

Table G: Mode 2

Value	Description

1	Fix not available
2	2D (<4 SVs used)
3	3D (>3 SVs used)

3.4.2.4.4. GSV - GNSS Satellites in View

This sentence reports the number of satellites (SV) in view, satellite ID numbers, elevation, azimuth and SNR value. There could be four satellites information per transmission so; if the number of satellites in view is bigger, separated GSV sentences will be generated. The number of sentence in transmission and the total to be transmitted is shown in the first 2 fields of the sentence. Table G contains the values for the following example:

$GPGSV,2,1,07,07,79,048,42,02,51,062,43,26,36,256,42,27,27,138,42,1*71

$GPGSV,2,2,07,09,23,313,42,04,19,159,41,15,12,041,42,1*41

Table H: GSV Data Format

Name	Example	Units	Description
Message ID	$GPGSV		GSV protocol header GP: GPS Talker ID GL: GLONASS Talker ID BD: BEIDOU Talker ID
Number of Messages	2		Range 1 to 3
Message Number1	1		Range 1 to 3
Satellites in View	07		
Satellite ID	07		Channel 1 GPS : 1-32 SBAS : 33-64 (offset 87) GLONASS : 65-96. GALILEO :1-36 (offset 300) BEIDOU :1-37 (offset 200)
Elevation	79	degrees	
Azimuth	048	degrees	
SNR (C/No)	42	dBHz	
....	....	....	

Satellite ID	27		Channel 4
Elevation	27	degrees	Channel 4 (Maximum 90)
Azimuth	138	degrees	Channel 4 (True, Range 0 to 359)
SNR (C/No)	42	dBHz	Range 0 to 99, null when not tracking
Signal ID	1		GPS, SBAS: 1 (L1 C/A); GLONASS: 1 (L1 C/A); GALILEO: 7(E1B/C); BEIDOU: 1(B1I)
Checksum	*71		
<CR> <LF>			End of message termination

3.4.2.4.5. RMC - Recommended Minimum Specific GNSS Data

This sentence reports Time, date, position, and course and speed data. Table H contains the values for the following example:

$GPRMC,161229.48,A,3723.247533,N,12158.341633,W,0.13,309.62,281118,6.1,W,A,V*10

Table I: RMC Data Format

Name	Example	Units	Description
Message ID	$GPRMC		RMC protocol header GP: GPS Talker ID
UTC Time	161229.48		hhmmss.ss
Status	A		A=data valid or V=data not valid
Latitude	3723.247533		ddmm.mmmmmm
N/S Indicator	N		N=north or S=south
Longitude	12158.341633		dddmm.mmmmmm
E/W Indicator	W		E=east or W=west
Speed Over Ground	0.13	knots	
Course Over Ground	309.62	degrees	True
Date	281118		ddmmyy
Magnetic Variation	6.1	degrees	E=east or W=west

Mag variation direction	W		E/W. E subtracts mag var from true, W adds mag var to true.
Mode Indicator	A		See Table D
Navigational status Indicator	V		V (equipment is not providing navigational status indication).
Checksum	*10		
<CR> <LF>			End of message termination

3.4.2.4.6. VTG - Course over Ground and Ground Speed

This sentence reports the actual course and speed relative to the ground.

Table I contains the values for the following example:

$GPVTG,309.62,T, ,M,0.13,N,0.2,K,A*23

Table J: VTG Data Format

Name	Example	Units	Description
Message ID	$GPVTG		VTG protocol header GP: GPS Talker ID
Course	309.62		Measured heading
Reference	T		True
Course		degrees	Measured heading
Reference	M		Magnetic
Speed	0.13	knots	Measured horizontal speed
Units	N	Knots	
Speed	0.2	km/hr	Measured horizontal speed
Units	K		Kilometers per hour
Mode Indicator	A		See Table D
Checksum	*23		
<CR> <LF>			End of message termination

3.4.2.4.7. GNS - GNSS fix data

This sentence reports the GNSS fix data.

Table I contains the values for the following example:

$GNGNS,084509.00,3731.283789,N,12655.755481,E,ANNN,07,1.2,110.7,18.0,,,V*26

Table K: GNS Data Format

Name	Example	Units	Description
Message ID	$GNGNS		GNS protocol header GN: GNSS Talker ID
UTC Time	084509.00		hhmmss.ss
Latitude	3731.283789		ddmm.mmmmmm
N/S Indicator	N		N=north or S=south
Longitude	12655.755481		dddmm.mmmmmm
E/W Indicator	E		E=east or W=west
Mode Indicator	ANNN	km/hr	Fixed length field; contains four characters, The first symbol relates to GPS The second one – to GLONASS The third one – to GALILEO The fourth one – to BEIDOU See Table D
Satellites Used	07		Number of satellites in use, (Gps+Glonass+Galileo+Beidou)
HDOP	1.2		Horizontal Dilution Of Precision.
MSL Altitude	110.7	meters	Antenna Altitude above/below mean-sea-level (geoid)
Geoid Separation	18.0		The difference between the WGS-84 earth ellipsoid and the mean-sea-level (geoid), "-" means mean-sea-level below ellipsoid.
Age of Diff. Corr.		second	Null fields when DGPS is not used
Diff. Ref. Station ID			Null fields when DGPS is not used / 0000-1023
Navigational Status	V		V (Equipment is not providing navigational status indication)
Checksum	*26		
<CR> <LF>			End of message termination

3.4.2.4.8. GRS - Range residuals

The sentence is used to support the Receiver Autonomous Integrity Monitoring (RAIM).

Table I contains the values for the following example:

$GPGRS,085634.00,0,-7.468427,-4.436067,-8.816755,-4.297600,-5.622384,-2.630362,,,,,,,1,1*49

Table L: GRS Data Format

Name	Example	Units	Description
Message ID	$GPGRS		GRS protocol header GP: GPS Talker ID
UTC Time	085634.00		hhmmss.ss
Mode	0		Mode: 0 = residuals used to calculate the position given in the GGA or GNSmessage 1 = residuals were recomputed after the GGA or GNS message position was computed.
Range residuals	-7.468427	meters	Range residuals, in meters, for satellites used in the navigation solution. Order must match order of satellite ID numbers in GSA message. When GRS message is used, the GSA and GSV messages are generally required with this message.
....	....	....	
Range residuals		meters	
GNSS System ID	1		1=GPS 2=GLONASS 3=GALILEO 4=BEIDOU
Signal ID	1		GPS, SBAS: 1 (L1 C/A); GLONASS: 1 (L1 C/A); GALILEO: 7(E1B/C); BEIDOU: 1(B1I)
Checksum	*49		
<CR> <LF>			End of message termination

3.4.2.5. Checking GNSS Device Functionality

After a proper power on, the device is ready to receive AT commands on the MODEM serial port.

When the **$GPSP** command is issued, The GNSS receiver also will be powered on and it will start the scan of the available GNSS signals.

On the NMEA USB port (default 115200 bps, 8, n, 1) there must be presence of the NMEA sentences when the **$GPSNMUN** command is issued.

3.4.2.6. Controlling GNSS Receiver

The LE910Cx module is provided by a set of AT commands that permits to configure and use it through the MODEM serial port.

3.4.2.6.1. Power Control of GNSS Receiver

The GNSS receiver is by default switched off at the first power on. If is necessary to switch it on or off is possible to use the AT$GPSP command. The GNSS receiver is usable if the module is switched on (or at least in power saving). This command also switches off the GNSS receiver supply.

Syntax of the command **AT$GPSP=<status>**

Where:

<status> - 0 GPS controller is powered down(default), 1 GPS controller is powered up? Returns the range of values accepted

AT$GPSP? will return the current status.

Example 1: (to switch on the GNSS)

AT$GPSP=1<CR>

OK

Example 2: (to know the status)

AT$GPSP?<CR>

The answer will be:

$GPSP: 0

OK

3.4.2.6.2. GNSS Reset

With the command **AT$GPSR=<reset_type>** is possible to reset the GNSS module.

Pa1rameter:

<reset_type>

0 - Factory reset: This option clears all GPS memory including clock drift. It is available in controlled mode only.

1 - Coldstart (No Almanac, No Ephemeris): this option clears all data that is currently stored in the internal memory of the GPS receiver including position, almanac, ephemeris, and time. The stored clock drift however, is retained. It is available in controlled mode only.

2 - Warmstart (No ephemeris): this option clears all initialization data in the GPS receiver and subsequently reloads the data that is currently displayed in the Receiver Initialization Setup screen. The almanac is retained but the ephemeris is cleared. It is available in controlled mode only.

3 - Hotstart (with stored Almanac and Ephemeris): the GPS receiver restarts by using the values stored in the internal memory of the GPS receiver; validated ephemeris and almanac. It is available in controlled mode only.

Example:

It is available in controlled mode only.

AT$GPSP=1<CR>

OK

Let's suppose to perform a cold start of the GNSS receiver.

AT$GPSR=1<cr>

OK

The Receiver will clear all the parameters in its memory and it will start a new scanning of the available satellites.

3.4.2.6.3. GNSS Parameters Save

This command allows saving the set parameters in the module's memory

Syntax of the command

AT$GPSSAV

3.4.2.6.4. Restore GNSS Parameters

This command allows restoring the factory default parameters for the GNSS module

Syntax of the command:

AT$GPSRST

After this command should restart the module to update the modifications.

If the GPS controller is powered up (see $GPSP), the GPS controller is powered down because the GPS parameters should be reset with factory default.

3.4.2.6.5. Read Acquired GNSS Position

This command allows reading the acquired position of the GNSS receiver

Syntax of the command

AT$GPSACP

The response syntax is:

$GPSACP:<UTC>,<latitude>,<longitude>,<hdop>,<altitude>,<fix>,<cog>,<spkm>,<spkn>,<date>,<nsat_gps>,<nsat_glonass>

The fields contain the following information:

<UTC> - UTC time (hhmmss.sss) referred to GGA sentence

<latitude> - format is ddmm.mmmm N/S (referred to GGA sentence)

where:

dd – degrees - 00..90

mm.mmmm - minutes - 00.0000..59.9999

N/S: North / South

<longitude> - format is dddmm.mmmm E/W (referred to GGA sentence)

where:

ddd - degrees - 000..180

mm.mmmm - minutes - 00.0000..59.9999

E/W: East / West

<hdop> - x.x - Horizontal Diluition of Precision (referred to GGA sentence)

<altitude> - xxxx.x Altitude - mean-sea-level (geoid) in meters (referred to GGA sentence)

<fix> -

0 or 1 -Invalid Fix

2 - 2D fix

3 - 3D fix

<cog> - ddd.mm - Course over Ground (degrees, True) (referred to VTG sentence)

where:

ddd - degrees - 000..360

mm – minutes - 00..59

<spkm> - xxxx.x Speed over ground (Km/hr) (referred to VTG sentence)

<spkn> - xxxx.x- Speed over ground (knots) (referred to VTG sentence)

<date> - ddmmyy Date of Fix (referred to RMC sentence)

where:

dd - day - 01..31

mm – month - 01..12

yy – year - 00..99 - 2000 to 2099

<nsat_gps> - nn - Total number of GPS satellites in use (referred to GGA sentence)

- 00..12

<nsat_glonass> - nn - Total number of GLONASS satellites in use

- 00..12

Example:

$GPSACP: 3206.4020N,03450.2678E,1.1,3.3,0,0.0,0.0,0.0,030613,06,03

OK

3.5. Packet Switched Data Operations

3.5.1. USB Tethering Connection

3.5.1.1. Dial-Up Networking

It is legacy method to access internet service using public switched telephone network. The DTE uses an attached modem to send and receive internet protocol packets. So, it is limited to support high speed data rate over LTE technology. Not recommend using this method for internet access

3.5.1.2. Standard ECM/RNDIS

ECM stands for Ethernet Control Model and is an Ethernet emulation protocol defined by the USB Implementers Forum. RNDIS (Remote Network Driver Interface Specification) is a Microsoft proprietary protocol used mostly on top of USB. It provides a virtual Ethernet link to most versions of the Windows and Linux operating system.

Most of all, USB is configured to support ECM or RNDIS by issuing AT#USBCFG command.

AT#USBCFG=<composition>

For ECM, <composition> is set 4.

For RNDIS, <composition> is set 1.

After executing the command, DUT automatically reboots and then Host reconfigures USB composition accordingly.

ECM session can be established by running AT#ECM command.

AT#ECM=<Cid>,<Did>[,<UserId>,[<Pwd>,[<DhcpServerEnable>]]]

On the other hand, RNDIS session can be set up by AT#RNDIS command

AT#RNDIS=<Cid>,<Did>[,<UserId>,[<Pwd>,[<DhcpServerEnable>]]]

Refer to AT command guide document for more information on ECM/RNDIS control commands

ECM and RNDIS provides private IP address to the tethered TE(Host PC) even if the module has network-assigned IP address and communicates with WWAN N/W using NATing.

3.5.1.3. MBIM/RmNet

MBIM is communication class subclass specification for Mobile broadband interface model. It is a protocol by which the USB hosts and mobile broadband devices can efficiently exchange control commands and data frames. MBIM extends the Network Control Model (NCM) as a protocol between the host and USB devices, with the difference that devices transfer raw IP packets instead of packets with 802.3 headers. The Mobile Broadband Interface Model (MBIM) class driver is an inbox driver provided my Microsoft; no third-party driver is required.

RmNet is a Qualcomm proprietary mobile broadband network interface, emulating network interface for the connected TE and allowing for the module to behave as a network adapter. RmNet relies on a control interface for any control signaling between the TE and MS to initiate a data session on demand and send any notifications. The control interface is called as QMI(Qualcomm MSM Interface).

USB needs to be configured to support these types of interfaces by issuing AT#USBCFG command.

AT#USBCFG=<composition>

For RmNet, <composition> is set 0.

For MBIM, <composition> is set 2.

After executing the command, DUT automatically reboots and then Host reconfigures USB composition accordingly.

There are no AT commands controlling data session over these N/W interfaces. Instead, customer needs to prepare for their own connection manager or open source solution on linux environment.

MBIM/RmNet provides network-assigned IP address to the tethered TE(Host PC) and module just plays a role of data modem without NATing

3.5.2. Socket AT Commands

3.5.2.1. Configuring embedded TCP/IP Stack

Use the #SCFG command to configure a socket belonging to the Multi-socket environment, the <connId> parameter identifies the socket. The Multi-socket environment provides N sockets, the N value depends on the module you are using. The configuration is saved in NVM.

The command syntax is:

AT#SCFG=<connId>,<cid>,<pktSz>,<maxTo>,<connTo>,<txTo>

Example)Check the current PDP contexts configuration.

AT+CGDCONT?

+CGDCONT: 1,"IP"," Access_Point_Name ","",0,0

OK

Check the current Multi-sockets/PDP contexts configuration.

AT#SCFG?

#SCFG: 1,1,300,90,600,50

#SCFG: 2,1,300,90,600,50

#SCFG: 3,1,300,90,600,50

#SCFG: 4,2,300,90,600,50

#SCFG: 5,2,300,90,600,50

#SCFG: 6,2,300,90,600,50

OK

Check if some PDP context is active. The following response shows that no PDP contexts are active.

AT#SGACT?

#SGACT: 1,0

OK

3.5.2.2. Activating PDP Context

The #SGACT command activates/deactivates one of the PDP contexts defined with +CGDCONT command.

The command syntax is:

AT#SGACT=<cid>,<stat>[,<userId>,<pwd>]

Example

We want to activate context number one defined with +CGDCONT.

AT#SGACT=1,1

#SGACT: “212.195.45.65”

OK

3.5.2.3. Online Mode Operation

3.5.2.3.1. Open socket connection

The #SD command (Socket Dial) opens the TCP/UDP connection towards the host. If required, DNS query is done to resolve the IP address. To open the remote connection, the PDP context to which the <connId> is associated must be active, otherwise the command returns an ERROR message.

The command syntax is:

AT#SD=<connId>,<txProt>,<rPort>,<IPaddr>[,<closureType>[,<lPort>[,<connMode>[,<txTime>[,<userIpType>]]]]]

Example)Open socket connection with <connId>=1 in ONLINE mode

AT#SD=1,0,80,”www.telit.com”

CONNECT

If the command is successful we’ll have a CONNECT message, and the socket number 1 will be connected to the Telit webserver.

From this moment the data incoming in the serial port is sent to the Internet host, while the data received from the host is serialised and flushed to the Terminal Equipment.

The +++ sequence does not close the socket, but only suspends it.

3.5.2.3.2. Resume suspended connection

Use #SO command to resume a suspended connection.

The command syntax is:

AT#SO=<connId>

Example)

AT#SD=1,0,80, ”www.telit.com”

CONNECT

...

(+++)

OK

SRING: 1

AT#SO=1

CONNECT

(+++)

3.5.2.3.3. Close the connection

Use #SH command to close a socket connection. The command returns the OK message if the connection is closed.

The command syntax is:

AT#SH=<connId>

Example)Open a socket connection.

AT#SD=1,0,80,”www.telit.com”

CONNECT

…

+++

OK

Type in #SH command to close the socket connection.

AT#SH = 1

OK

3.5.2.4. Command Mode Operation

3.5.2.4.1. Open socket connection

Use #SD command with <connMode>=1 to open a connection in COMMAND mode.

<connMode> is the last parameter in the syntax command.

AT#SD=<connId>,<txProt>,<rPort>,<IPaddr>[,<closureType>[,<lPort>[,<1>[,<txTime>[,<userIpType>]]]]]

Example)Open socket connection <connId>=1 in COMMAND mode

AT#SD=1,0,80,”www.telit.com”,0,0,1

OK

3.5.2.4.2. Send user data

Use #SSEND command to send data on connection when the module is in COMMAND mode. When the <CR> is entered to close the entering of the command, the ">" prompt appears to indicate that the command is ready to accept the data to be sent.

Enter Ctrl-Z to close the data entering and send the data. Before using the command, the socket must be opened.

The command syntax is:

AT#SSEND=<connId>

Example)Send the string “hello” on an echo socket with SRING mode set to Data amount.

AT#SSEND=1

> hello<CTRL-Z>

OK

SRING: 1,5

Use #SSENDEXT command to include all bytes (0x00 to 0xFF) in the block of data to send. This command allows to include special characters as ESC (0x1B), Ctrl-Z (0x1A), BS (0x08) not accepted by #SSEND. The command syntax is:

AT#SSENDEXT=<connId>,<bytestosend>

3.5.2.4.3. Receive user data

The module is in COMMAND mode, and assume to have received a SRING unsolicited indicator notifying that received data are pending in the socket. Use #SRECV command to get the pending data in the socket buffer. The command syntax is:

AT#SRECV=<connId>,<maxByte>[,<UDPInfo>]

Example)

We receive a SRING data amount and then we extract all the five bytes pending with SRECV.

SRING: 1,5

AT#SRECV=1,5

#SRECV: 1,5

hello

OK

3.5.2.4.4. Close the connection

The #SH (Socket Shutdownl) command are introduced before

3.5.2.5. Server Mode(Socket Listen) Operation

3.5.2.5.1. Create listen socket

Use the #SL (Socket Listening) command to open a socket in listening mode for an incoming TCP connection.

When a remote host tries to connect, the module sends to the DTE the +SRING: <connId> unsolicited indication. The user can accept (#SA) or refuse (#SH) the incoming connection.

The command syntax is:

AT#SL=<connId>,<listenState>,<listenPort>[,<lingerT>]

Example)Open the <connId> = 1 socket in listening mode on <port> = 6543.

AT#SL=1,1,6543

OK

3.5.2.5.2. Accept incoming connection

Use the #SA command without <connMode> parameter to accept the incoming connection, notified by the SRING unsolicited indication, in ONLINE mode.

The command syntax is:

AT#SA=<connId>[,<connMode>]

Example)

Now, if a remote host tries to connect, the module receives a SRING unsolicited indication with the listening <connId>:

SRING: 1

Accept the incoming connection <connId>=1 in ONLINE mode.

AT#SA=1

CONNECT

… exchange data …

The module is in ONLINE mode, the connection is established and the two hosts can exchange data. With the escape sequence (+++) the connection can be suspended, and the module is back to COMMAND mode.

3.5.2.5.3. Command Mode Operation

Use the #SA command with <connMode>=1 to accept the incoming connection in COMMAND mode.

The command syntax is:

AT#SA=<connId>[,<connMode>]

Example)

Now, if a remote host tries to connect, the module receives a SRING unsolicited indication with the listening <connId>:

SRING: 1

Accept the incoming connection <connId>=1 in ONLINE mode.

AT#SA=1,1

OK

3.5.3. SSL AT Commands

TLS and its predecessor SSL are cryptographic protocols used over the Internet to provide secure data communication in several applications.

For TLS protocol, see standards:

- RFC 2246 - TLS Protocol Version 1.0
- RFC 4346 - TLS Protocol Version 1.1
- RFC 5246 - TLS Protocol Version 1.2

3.5.3.1. Configuration

To provide communication security over a channel, enable a SSL socket using the #SSLEN command.

AT#SSLEN= <SSId>,<Enable>

Ex) Enable SSL functionality,

AT#SSLEN=1,1

OK

Use the following command to select the protocol.

AT#SSLSECCFG2=<SSId>,<version>
[,<unused_A>[,<unused_B>[,<unused_C>[,<unused_D>]]]]

Ex) Select TLS1.2 protocol,

AT#SSLSECCFG2=1,2

OK

The cipher suite is the set of algorithms used to negotiate the security settings for a network connection using the SSL/TLS network protocol.

AT#SSLSECCFG=<SSId>,<CipherSuite>,<auth_mode>[,<cert_format>]

Ex) Set SSL configuration,

AT#SSLSECCFG=1,0,1,1

OK

The following types of security data can be stored in the modules:

- Certificates
- CA Certificates
- Private Key

AT#SSLSECDATA=<SSId>,<Action>,<DataType>[,<Size>]

Ex) Store CA certificate,

AT#SSLSECDATA=1,1,1,<size>

> -----BEGIN CERTIFICATE-----<LF>

[…]

-----END CERTIFICATE-----<LF>

<ctrl>Z

OK

Use the following command to configure the SSL socket, before opening it.

AT#SSLCFG=<SSId>,<cid>,<pktSz>,<maxTo>,<defTo>,<txTo>[<sslSRingMode>[<noCarrierMode>[,<skipHostMismatch >[,<UNUSED_4>]]]]

Ex) Set SSL configuration,

AT#SSLCFG=1,1,300,90,100,50,0,0,1,0

OK

Refer to AT Commands Reference Guide for more details.

3.5.3.2. Online Mode Operation

3.5.3.2.1. Open secure socket connection

Use the following command to open a SSL socket

AT#SSLD=<SSId>,<rPort>,<IPAddress>,<ClosureType>[,<connMode>[,<Timeout>]]

Ex) Open the SSL socket in ONLINE mode,

AT#SSLD=1,443,"123.124.125.126",0,0

CONNECT

... [Bidirectional data exchange] ...

+++ [suspend the connection]

OK

On success, the CONNECT message is returned, and from now all bytes sent to the serial port are forwarded to the remote server.

It is possible suspend the connection, without closing it, by sending the escape sequence (+++). After that, the module returns the OK response and can parse again AT commands.

ONLINE mode can be restored at any time by sending the following command.

AT#SSLO=<SSId>

Ex) Restore from Command Mode to Online Mode,

AT#SSLO=1

CONNECT

3.5.3.2.2. Exchanging user data

Refer to the opening of socket connection on online mode for AT command.

Open the SSL socket and wait CONNECT message. After receiving the CONNECT message, you can send/receive data to the module. Data are encrypted and sent to the server through the secure socket as soon as the packet size has been reached or the <txTo> timeout expires

In ONLINE mode, you cannot enter AT commands on the used serial port

3.5.3.2.3. Close the connection

The following command closes the SSL socket.

AT#SSLH=<SSId>[,<ClosureType>]

If the secure socket has been opened in ONLINE mode, the user needs to send the escape sequence (+++) before closing it with #SSLH command, unless the communication is remotely closed or the idle inactivity timeout expires (NO CARRIER message).

If the secure socket has been opened in COMMAND mode, when communication is remotely closed and all data have been retrieved (#SSLRECV), you can also close on client side and NO CARRIER message is displayed. At any moment, it is also possible to close the secure socket on client side by means of #SSLH.

Ex) Close SSL connection,

AT#SSLH=1

OK

3.5.3.3. Command Mode Operation

3.5.3.3.1. Open secure socket connection

In COMMAND mode, data can be exchanged through a SSL socket by means of the #SSLSEND, #SSLSENDEXT and #SSLRECV commands.

Use the following command to open a SSL socket

AT#SSLD=<SSId>,<rPort>,<IPAddress>,<ClosureType>[,<connMode>[,<Timeout>]]

Ex) Open with Command Mode,

AT#SSLD=1,server_port,”server_address”,0,1,100

OK

Use the following command to query the status of SSL socket

AT#SSLS=<SSId>

Ex) Query the status of the Secure Socket Id = 1,

AT#SSLS=1

#SSLS: 1,2,<cipher_suite>

OK

Use the following command to get information about SSL socket data traffic

AT#SSLI[=<SSId>]

Ex) Get SSL socket information

AT#SSLD=1,server_port,"server_address",0,1

OK

SSLSRING: 1,16384

AT#SSLI=1

#SSLI: 1,0,0,16384,0

OK

3.5.3.3.2. Send user data

Use one of the following commands to send data:

AT#SSLSEND=<SSId>[,< Timeout >]

When the command is closed with a <CR>, the ‘>’ prompt appears. Now, you can enter the data to be sent. To close the data block, enter <ctrl>Z, then the data are forwarded to the remote server through the secure socket. Response: OK on success, ERROR on failure.

Ex) Send data,

AT#SSLSEND=1,100

> (send data)

<ctrl>Z

OK

AT#SSLSENDEXT=<SSId>,<bytestosend>[,<Timeout>]

When the command is closed with <CR>, the '>' prompt appears. Now, you can enter the data to be sent. When <bytestosend> bytes have been sent, operation is automatically completed. Response: OK on success, ERROR on failure.

Ex) Send data,

AT#SSLSENDEXT=1,100,100

> (send data)

OK

3.5.3.3.3. Receive user data

Data can be received in two different ways:

- Using the #SSLRECV command (the "standard" way),
- Reading data from the SSLSRING: unsolicited message.

Use the following command to receive data.

AT#SSLRECV=<SSId>,<MaxNumByte>[,<TimeOut>]

On success, the data are displayed in the following format:

#SSLRECV: <numBytesRead>

… received data ….

OK

If the timeout expires, the module displays the following response

#SSLRECV: 0

TIMEOUT

OK

The ERROR message appears on failure.

The SSLSRING: unsolicited message, if enabled, notifies the user of any new incoming data.

Configuring <sslSRingMode>=2 by means of the #SSLCFG command data is displayed in the URC in this format:

SSLSRING:<SSId>,<dataLen>,<data>

3.5.3.3.4. Close the connection

Refer to closure operation on online mode.

3.5.4. HTTP AT Commands

3.5.4.1. Configuration

HTTP parameters should be configured using the following AT command before running HTTP operations.

AT#HTTPCFG=<prof_id>,<server_address>,<server_port>,<auth_type>,<username>,<password>,<ssl_enabled>,<timeout>,<cid>,<pkt_size>

Ex) Set HTTP configuration,

AT#HTTPCFG=0,server_address,server_port,1,username,password,0,120,1,1

OK

3.5.4.2. Query

Using the following command, it is possible to send a HTTP GET, HEAD or DELETE operation to HTTP server.

AT#HTTPQRY=<prof_id>,<command>,<resource>,<extra_header_line>

Ex) Query,

AT#HTTPQRY=0,0,/MOTDATA_N_LONG.txt

OK

AT#HTTPQRY=0,0,"/client_info.php","user-agent:LE910Cx"

OK

If sending is done successfully, the response is OK. Otherwise an error code is reported.

When the HTTP server answer is received, this product sends the following URC through serial interface (USB or UART)

#HTTPRING:<prof_id>,<http_status_code>,<content_type>,<data_size>

3.5.4.3. Request

Using the following command, it is possible to send a HTTP POST or PUT operation to HTTP server.

AT#HTTPSND=<prof_id>,<command>,<resource>,<data_len>,<post_param>,<extra_header_line>

If sending is done successfully, the response is OK. Otherwise an error code is reported.

When the HTTP server answer is received, the module sent the URC(refer to #HTTPRING URC)

Note : When using the AT#HTTPSND command, the HTTP header always include "Connection: close" line and it cannot be removed.

Ex) Send data,

AT#HTTPSND=1,0,"/upload/110_01.txt",110

>>> (send data)

OK

AT#HTTPSND=1,0,"/cgi-bin/upload.php",343,"3:boundary=----WebKitFormBoundarylejjShOaaqpRD1dO"

>>> (send data)

OK

3.5.4.4. Receive

When being notified with #HTTPRING URC, it is possible to read data from HTTP server using the AT#HTTPRCV command.

AT#HTTPRCV=<prof_id>,<maxByte>

Ex) Receive data,

AT#HTTPRCV=0,0

….

OK

3.5.5. FTP AT Commands

3.5.5.1. Configuration

FTP parameters can be configured using following AT command. Before running FTP, it is required that customer changes FTP parameters if it is needed.

AT#FTPCFG=<tout>,<IPPignoring>,<FTPSEn>,<FTPext>

AT#FTPTO=[=<tout>]

Ex)

Configure FTP parameters

AT#FTPCFG=500,0,0,1

OK

AT#FTPTO=1000

OK

3.5.5.2. Open connection

To take advantage of FTP capability, it is required that FTP connection has to first of all be opened to the server. Afterwards, FTP commands could be available on the connected FTP channel.

Ex)

AT#FTPOPEN=<server:port>,<username>,<password>,<mode>

If it is failed to establish FTP connection within Time-out set by AT#FTPCFG, #FTPOPEN command stops and return error message to DTE.

Ex)Open FTP connection in active mode

AT#SGACT=1,1

#SGACT: 174,156,82,131

OK

AT#FTPOPEN="server","username","password",0

OK

3.5.5.3. Online mode operation

3.5.5.3.1. Uploading

This example shows how to upload a file to a FTP server when the module is in ONLINE mode. A terminal emulator is connected to the module.

Ex)

AT#SGACT=1,1

#SGACT: 193.199.234.255

OK

Set the FTP time-out.

AT#FTPTO=1000

OK

FTP connection open and in active mode.

AT#FTPOPEN="server","username","password",0

OK

The following #FTPPUT command opens the data connection, and the module enters ONLINE mode. "filename.txt" is the file name where the data will be stored on the FTP server. If the file you are sending is a text file, the extension must be .txt.

AT#FTPPUT="filename.txt",0

CONNECT

⋯ type in the data to write in the filename.txt file stored on the FTP server ⋯

+++

NOCARRIER

Close FTP control connection.

AT#FTPCLOSE

OK

Deactivate the PDP context.

AT#SGACT=1,0

OK

3.5.5.3.2. Appending

If the file must be appended, use the AT#FTPAPP command (with <connMode> = 0) instead of #FTPPUT.

Except for changing from #FTPPUT to #FTPAPP, the procedures are the same as "Uploading"

3.5.5.3.3. Downloading

The procedure up to #FTPOPEN is the same as in "Uploading"

And then,

Ex)

Check the working directory.

AT#FTPPWD

#FTPPWD: 257 "/"

OK

Use #FTPLIST command to get the list of files on the working directory

Use the #FTPGET command to open e data connection, and download a file from the FTP server.

AT#FTPGET="filename.txt"

CONNECT

⋯ the content of the file appears on terminal emulator ⋯

NO CARRIER

Close FTP control connection.

AT#FTPCLOSE

OK

3.5.5.4. Command mode operation

3.5.5.4.1. Uploading

This example shows how to upload data toward a FTP server when the module is in COMMAND mode. A terminal emulator is connected to the module.

The procedure before #FTPOPEN is the same as "Uploading"

Ex)

AT#FTPOPEN="server","username","password",1

OK

The #FTPPUT command opens the data connection, and the module remains in COMMAND mode. filename.txt is the file name where the data will be stored on the FTP server.

AT#FTPPUT="filename.txt",1

OK

Enter the #FTPAPPEXT command to upload data. After entering <CR>, the command returns the ">" prompt. Now, enter the data to be sent to the FTP server. As soon as <bytestosend> bytes are written, data are sent to the FTP server, and the #FTPAPPEXT message is returned.

AT#FTPAPPEXT=bytestosend

>⋯ type in data⋯

#FTPAPPEXT: <SentBytes>

OK

Use again the #FTPAPPEXT=bytestosend command to send a new data chunk.

To send the last data chunk and close the FTP connection, use the following:
AT#FTPAPPEXT=bytestosend,1

3.5.5.4.2. Appending

If the file must be appended, use the AT#FTPAPP command (with <connMode> = 1) instead of #FTPPUT.

Except for changing from #FTPPUT to #FTPAPP, the procedure before and after is the same as "Uploading"

3.5.5.4.3. Downloading

Assume that the FTP connection is open. Use the #FTPGETPKT command to open a data connection, and download a file from the FTP server. Data are buffered on the socket; the module remains in COMMAND mode.

Ex)

AT#FTPGETPKT="filename.txt"

OK

The following command reports the number of bytes buffered on the socket.

AT#FTPRECV?

#FTPRECV: 600

OK

Read the first 400 bytes of the available buffered data.

AT#FTPRECV=400

#FTPRECV: 400

Text row number 1 * 111111111111111111111111 *

...

Text row number 8 * 88888888888888888

OK

Read 200 bytes – if available – starting from the position + 1 of the last byte read with the previous #FTPRECV command.

AT#FTPRECV =200

#FTPRECV: 200

88888 *

Text row number 9 * 999999999999999999999999 *

...

Text row number 12 * CCCCCCCCCCCCCCC

OK

The #FTPGETPKT? read command reports the download state.

AT#FTPGETPKT?

#FTPGETPKT: filename.txt,0,1

OK

The first parameter is the file name, the second indicates text or hex mode. The third parameter indicates <EOF> (End of File): 0 file transfer is in progress; 1 file transfer is ended.

3.5.5.5. Close the connection

If FTP connection is no longer needed, it can be disconnected using below command

AT#FTPCLOSE

OK

3.5.6. Email AT Commands

3.5.6.1. Configuration

It is required that customer configure this module with Email parameters prior to sending email data using this product. Following is sequence of AT commands required in configuring Email parameters

Configure SMTP server address used for Email sending

AT#ESMTP=<smtp>

Configure a sender address to be seen as originator of this email

AT#EADDR=<e-addr>

Configure email user id authenticated by SMTP server

AT#EUSER=<e-user>

Configure email password authenticated by SMTP server

AT#EPASSW=<e-pwd>

If needed, save those parameters into the module permanently

AT#ESAV

3.5.6.2. Sending an email

With assumption of Email parameters configured and PDP cid 1 activated by #SGACT, it is ready to send email to a specified receiver.

AT#EMAILD=<da>,<subj>

The device responds to the command with the prompt '>' and waits for the message body text. To complete the operation, send Ctrl-Z char (0x1A hex); to exit without writing the message, send ESC char (0x1B hex)

Ex)

Send the e-mail to the recipient having recipient_address.

AT#EMAILD="recipient_address","email subjext"

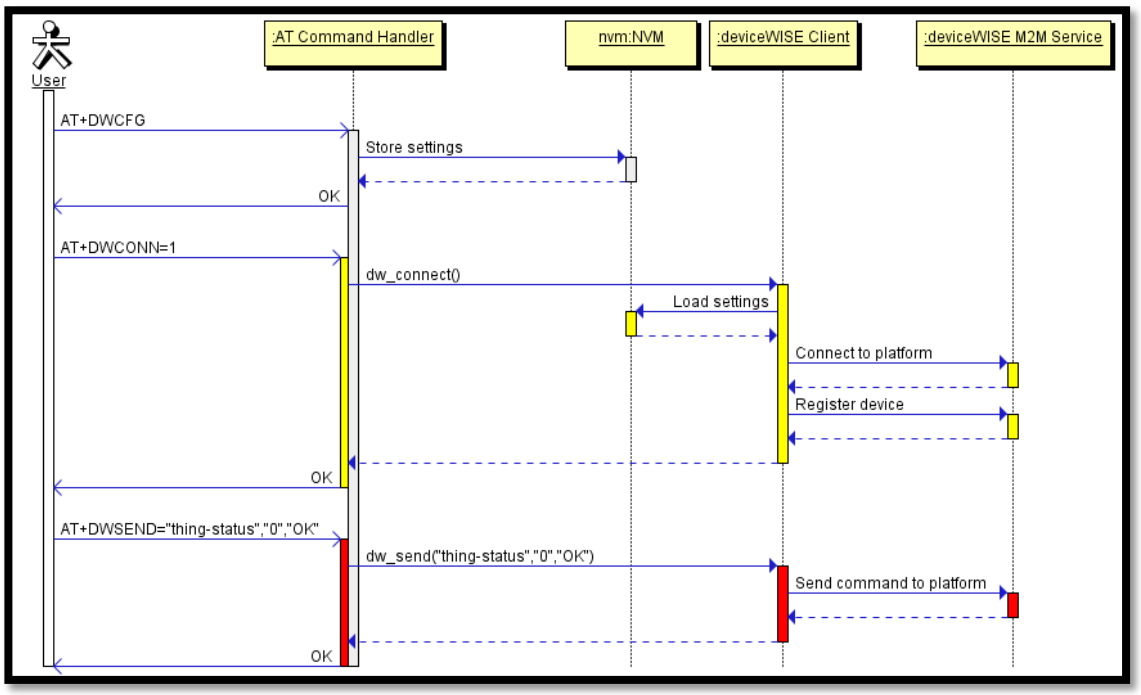
> Hello<Ctrl-Z>

OK

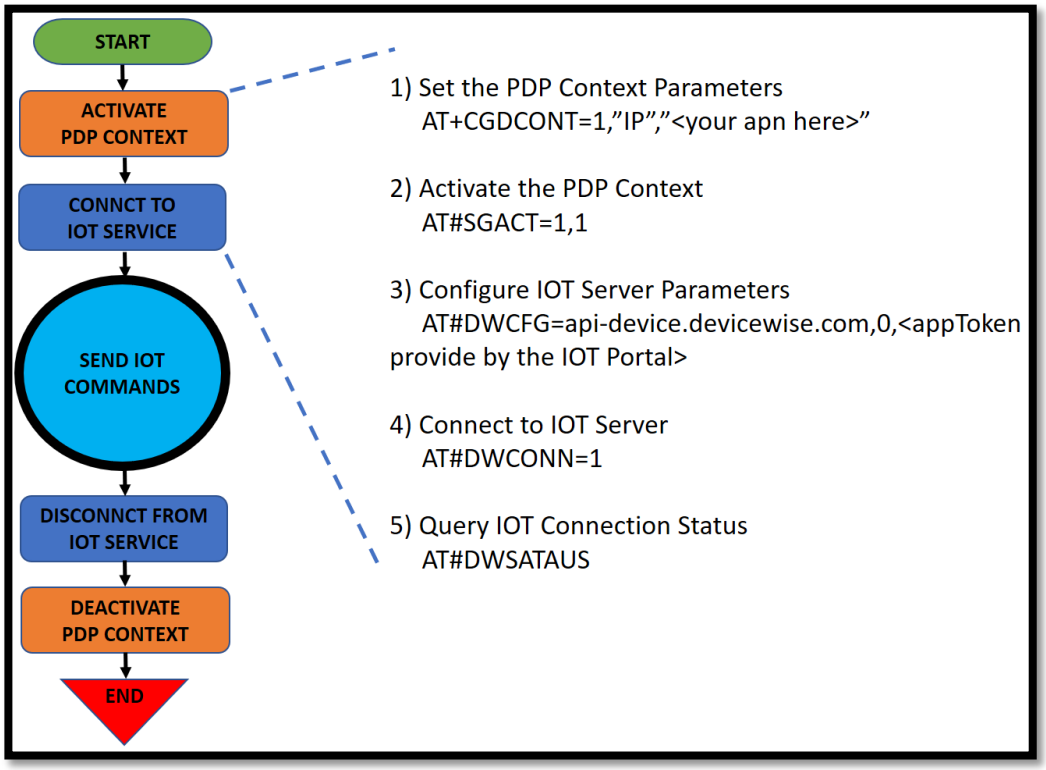
3.5.7. IOT Platform AT Commands

The LE910Cx module has been updated to include native support for interfacing applications to the M2M Service. These APIs allow you to use a new set of AT commands to easily expand the capabilities of any device built upon a LE910Cx module.

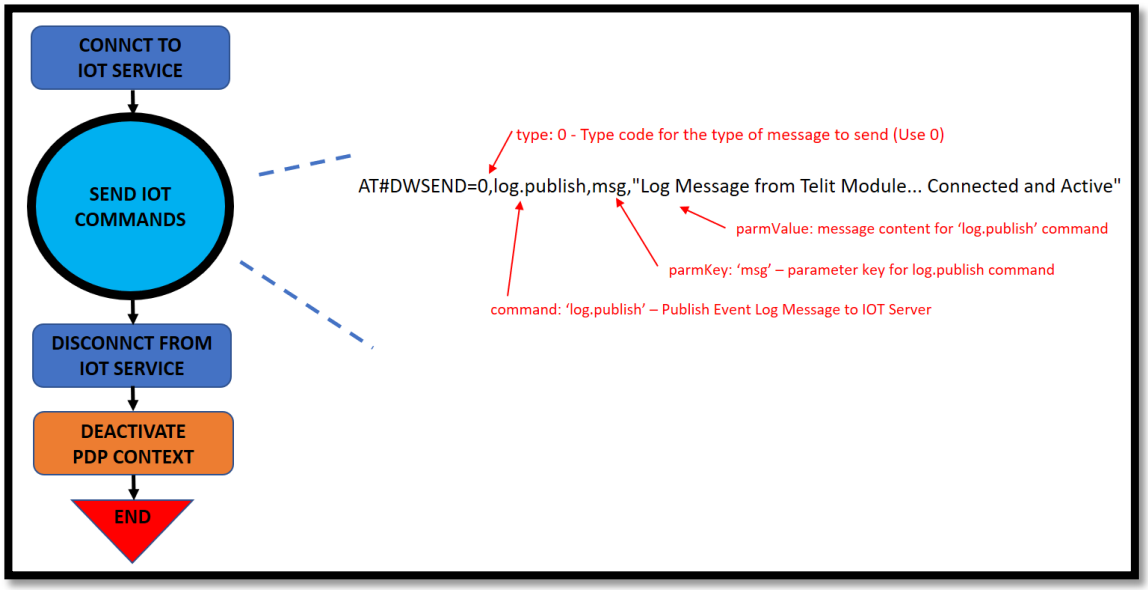
Here is a basic interaction diagram



3.5.7.1. Connect to IOT service



3.5.7.2. API and AT commands



API: property.publish

The property.publish command is used to publish property data (typically sensor data) for a thing.

```
## Publish a property called "sensor1" with a value of 123
## mything is automatically set.
AT#DWSEND=0,property.publish,key, myprop, value,123
#DWSEND: 3
OK
## Wait for a #DWRING notification to indicate we have a reply to our
API call.
#DWRING: 1,3,2
```

TR50 Request

```
{
  "cmd": {
    "command": "property.publish",
    "params": {
      "thingKey": "mything",
      "key": "myprop",
      "value": 123.44,
      "ts": "2018-11-05T02:03:04.322Z",
      "corrId": "mycorrid"
    }
  }
}
```

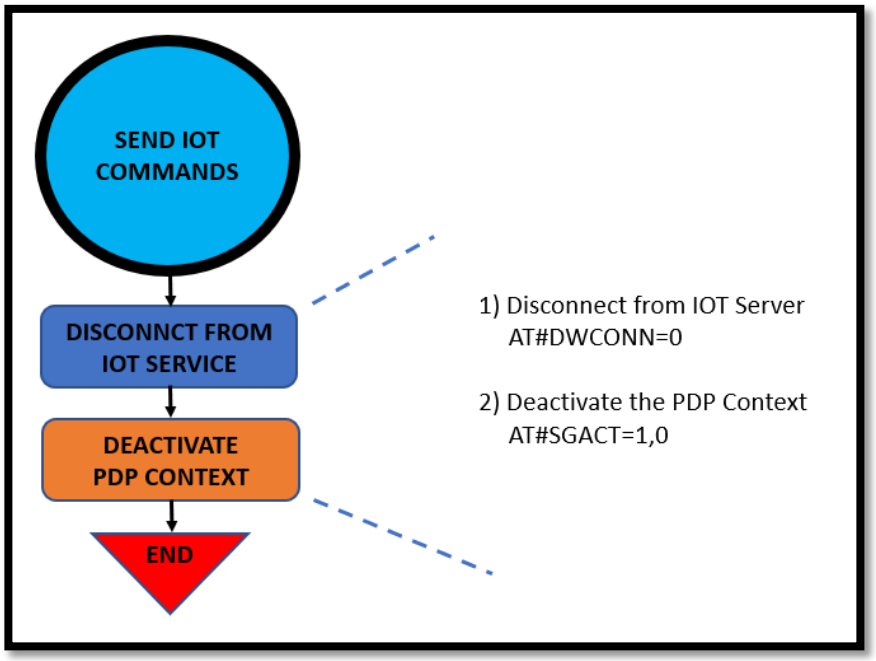
```
## Execute AT#DWRCV command to check the reply, note the "3"
matches the return from the #DWSEND command above.
## The "0" (second parameter in the response) indicates a success.
AT#DWRCV=3
OK
#DWDATA: 3,0,2,OK
```

TR50 Response

If the command is sent successfully a success message is returned.

```
{
  "cmd": {
    "success": true
  }
}
```

3.5.7.3. Disconnect from IOT Service



3.5.7.4. IOT Platform AT commands list

AT#DWCFG – configure deviceWISE parameters

AT#DWCONN – connect to deviceWISE server

AT#DWSTATUS – query connection status

AT#DWSEND – send data to deviceWISE server

AT#DWSENDR – send raw data to deviceWISE server

AT#DWRCV – receive data from deviceWISE server

AT#DWRCVR – receive raw data from deviceWISE server

AT#DWLRCV – List information on messages pending from deviceWISE server

AT#DWEN - Set command permits to enable/disable up to 8 different deviceWISE features.

NOTE: Please refer to the AT Commands Reference Guide document for more details.

4. PERFORMANCE MEASUREMENTS

4.1. Interrupt latencies

The interrupt latency is considered in this example using the time from the wakeup to the time the context switch happened.

Results were taken on a performance build. First table shows the results when the CPU governor is set to interactive (default settings) and the second table is while setting the governor to performance.

Task	Runtime ms	Switches	Average delay ms	Maximum delay ms	Maximum delay at
hwrng:50	0.071 ms	1	avg: 6.756 ms	max: 6.756 ms	max at: 4760.939225 s
kworker/u2:3:22	0.428 ms	2	avg: 0.131 ms	max: 0.159 ms	max at: 4768.610426 s
init:1	0.787 ms	2	avg: 0.121 ms	max: 0.160 ms	max at: 4765.410387 s
kworker/u2:7:903	0.705 ms	3	avg: 0.100 ms	max: 0.140 ms	max at: 4765.470639 s
msm_watchdog:13	0.000 ms	1	avg: 0.088 ms	max: 0.088 ms	max at: 4769.700320 s
qmi_linux_clnt:1168	0.245 ms	1	avg: 0.084 ms	max: 0.084 ms	max at: 4767.834634 s
kworker/0:13:2036	3.350 ms	15	avg: 0.079 ms	max: 0.254 ms	max at: 4769.850374 s
fota_redbend:1808	1.096 ms	4	avg: 0.075 ms	max: 0.111 ms	max at: 4764.300343 s
diagrebootapp:1543	2.303 ms	10	avg: 0.074 ms	max: 0.094 ms	max at: 4768.601101 s
dlt-daemon:1167	3.987 ms	11	avg: 0.073 ms	max: 0.105 ms	max at: 4767.550464 s
rcu_preempt:7	0.689 ms	6	avg: 0.065 ms	max: 0.092 ms	max at: 4760.960225 s
ubifs_bgt0_4:362	2.176 ms	9	avg: 0.059 ms	max: 0.111 ms	max at: 4765.460356 s
ksoftirqd/0:3	0.207 ms	3	avg: 0.058 ms	max: 0.080 ms	max at: 4767.550416 s
sleep:2046	8.904 ms	2	avg: 0.049 ms	max: 0.078 ms	max at: 4770.939396 s
cfinteractive:133	0.000 ms	2	avg: 0.029 ms	max: 0.032 ms	max at: 4760.990169 s
perf:2045	1.406 ms	1	avg: 0.029 ms	max: 0.029 ms	max at: 4770.940515 s
QCMAP_Connectio:817	0.799 ms	4	avg: 0.024 ms	max: 0.033 ms	max at: 4767.520488 s
TOTAL:	27.152 ms	77			

Figure 3: Results for interactive governor

Task	Runtime ms	Switches	Average delay ms	Maximum delay ms	Maximum delay at
hwrng:50	0.038 ms	1	avg: 0.492 ms	max: 0.492 ms	max at: 4845.200305 s
QCMAP_Connectio:817	0.086 ms	1	avg: 0.170 ms	max: 0.170 ms	max at: 4849.760414 s
rcu_preempt:7	0.393 ms	4	avg: 0.107 ms	max: 0.260 ms	max at: 4845.220644 s
dnsmasq:1518	0.205 ms	1	avg: 0.071 ms	max: 0.071 ms	max at: 4849.760289 s
ksoftirqd/0:3	0.519 ms	4	avg: 0.065 ms	max: 0.111 ms	max at: 4845.200338 s
diagrebootapp:1543	1.479 ms	10	avg: 0.058 ms	max: 0.094 ms	max at: 4845.620221 s
init:1	0.409 ms	2	avg: 0.056 ms	max: 0.062 ms	max at: 4845.490228 s
kworker/u2:3:22	0.201 ms	2	avg: 0.053 ms	max: 0.054 ms	max at: 4852.820223 s
fota_redbend:1808	0.648 ms	4	avg: 0.052 ms	max: 0.062 ms	max at: 4845.370228 s
qmi_linux_clnt:1168	0.137 ms	1	avg: 0.050 ms	max: 0.050 ms	max at: 4847.836400 s
dlt-daemon:1167	1.594 ms	10	avg: 0.049 ms	max: 0.049 ms	max at: 4854.549941 s
msm_watchdog:13	0.000 ms	1	avg: 0.044 ms	max: 0.044 ms	max at: 4849.860205 s
kworker/u2:7:903	0.280 ms	3	avg: 0.043 ms	max: 0.045 ms	max at: 4854.760232 s
sleep:2049	8.432 ms	2	avg: 0.037 ms	max: 0.049 ms	max at: 4855.207040 s
kworker/0:13:2036	1.339 ms	13	avg: 0.034 ms	max: 0.050 ms	max at: 4849.943542 s
ubifs_bgt0_4:362	1.314 ms	9	avg: 0.033 ms	max: 0.063 ms	max at: 4849.740229 s
perf:2048	8.859 ms	1	avg: 0.017 ms	max: 0.017 ms	max at: 4855.207575 s
TOTAL:	25.933 ms	69			

Figure 4: Results for performance governor

4.2. Memory bandwidth & Latencies

The test was made using the tiny membench utility which tries to measure the peak bandwidth of sequential memory accesses and the latency of random memory accesses. Bandwidth is measured by running different assembly code for the aligned memory blocks and attempting different prefetch strategies.

Bandwidth tests:

Results were taken on a performance build.

Note 1: 1MB = 1000000 bytes

Note 2: Results for 'copy' tests show how many bytes can be copied per second (adding together read and written bytes would have provided twice higher numbers)

Note 3: 2-pass copy means that we are using a small temporary buffer to first fetch data into it, and only then write it to the destination (source -> L1 cache, L1 cache -> destination)

C copy backwards	278.3 MB/s
C copy backwards (32 byte blocks)	814.5 MB/s
C copy backwards (64 byte blocks)	884.5 MB/s
C copy	847.1 MB/s
C copy prefetched (32 bytes step)	859.8 MB/s
C copy prefetched (64 bytes step)	884.9 MB/s
C 2-pass copy	783.7 MB/s
C 2-pass copy prefetched (32 bytes step)	809.1 MB/s
C 2-pass copy prefetched (64 bytes step)	832.1 MB/s
C fill	2313.4 MB/s
C fill (shuffle within 16 byte blocks)	2313.4 MB/s
C fill (shuffle within 32 byte blocks)	532.2 MB/s
C fill (shuffle within 64 byte blocks)	524.2 MB/s
standard memcpy	649.4 MB/s
standard memset	2314.6 MB/s
NEON read	1551.0 MB/s
NEON read prefetched (32 bytes step)	1683.8 MB/s
NEON read prefetched (64 bytes step)	1773.6 MB/s
NEON read 2 data streams	446.1 MB/s
NEON read 2 data streams prefetched (32 bytes step)	840.9 MB/s
NEON read 2 data streams prefetched (64 bytes step)	887.2 MB/s
NEON copy	875.3 MB/s
NEON copy prefetched (32 bytes step)	895.2 MB/s
NEON copy prefetched (64 bytes step)	962.1 MB/s

NEON unrolled copy	871.8 MB/s
NEON unrolled copy prefetched (32 bytes step)	913.2 MB/s
NEON unrolled copy prefetched (64 bytes step)	928.5 MB/s
NEON copy backwards	882.0 MB/s
NEON copy backwards prefetched (32 bytes step)	897.6 MB/s
NEON copy backwards prefetched (64 bytes step)	972.1 MB/s
NEON 2-pass copy	865.1 MB/s
NEON 2-pass copy prefetched (32 bytes step)	909.6 MB/s
NEON 2-pass copy prefetched (64 bytes step)	929.5 MB/s
NEON unrolled 2-pass copy	769.5 MB/s
NEON unrolled 2-pass copy prefetched (32 bytes step)	802.3 MB/s
NEON unrolled 2-pass copy prefetched (64 bytes step)	837.3 MB/s
NEON fill	2313.5 MB/s
NEON fill backwards	2313.5 MB/s
VFP copy	866.4 MB/s
VFP 2-pass copy	788.6 MB/s
ARM fill (STRD)	2313.0 MB/s
ARM fill (STM with 8 registers)	2314.2 MB/s
ARM fill (STM with 4 registers)	2314.0 MB/s
ARM copy prefetched (incr pld)	933.2 MB/s
ARM copy prefetched (wrap pld)	907.6 MB/s

Figure 6 - Memory bandwidth test results

Memory latency test

Average time is measured for random memory accesses in the buffers of different sizes. The larger is the buffer, the more significant are relative contributions of TLB, L1/L2 cache misses and SDRAM accesses. For extremely large buffer sizes we are expecting to see page table walk with several requests to SDRAM for almost every memory access (though 64MiB is not nearly large enough to experience this effect to its fullest).

Note 1: All the numbers are representing extra time, which needs to be added to L1 cache latency.

Note 2: Dual random read means that we are simultaneously performing two independent memory accesses at a time. In the case if the memory subsystem can't handle multiple outstanding requests, dual random read has the same timings as two single reads performed one after another.

block size	single random read / dual random read
1024	0.0 ns / 0.0 ns

2048	0.0 ns	/	0.0 ns
4096	0.0 ns	/	0.0 ns
8192	0.0 ns	/	0.0 ns
16384	0.0 ns	/	0.0 ns
32768	0.0 ns	/	0.0 ns
65536	4.8 ns	/	8.3 ns
131072	7.5 ns	/	11.7 ns
262144	9.9 ns	/	14.6 ns
524288	76.8 ns	/	122.8 ns
1048576	114.8 ns	/	163.5 ns
2097152	139.0 ns	/	184.0 ns
4194304	151.4 ns	/	193.4 ns
8388608	159.2 ns	/	200.5 ns
16777216	167.7 ns	/	212.5 ns
33554432	180.6 ns	/	235.5 ns

5. SERVICE AND FIRMWARE UPDATE

5.1. Firmware Update

The Telit Modules firmware is updated through the USB Interface normally used for the AT Commands.

It is suggested to provide an USB interface on the User Printed Circuit Board (where the Telit Module is soldered) to perform the physical connection between the Telit module and a Windows-based PC. That simple circuitry makes the firmware updating easy when a new firmware version is released.

During the User Application development or evaluation phase of the Telit module, the USB port implemented on the **Telit Evaluation Board (Telit EVB)** can be used to connect the Telit module to a Windows-based PC on which a dedicated tool for firmware updating is running.

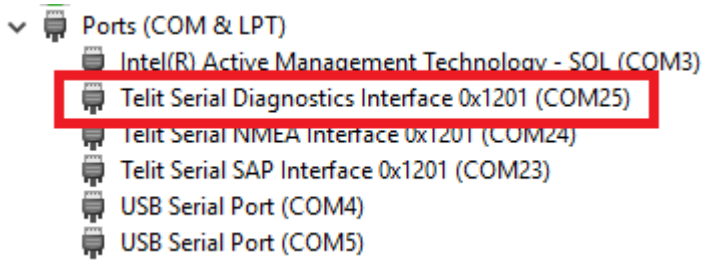
Telit provides the User with two tools to update the firmware of the module. The following paragraphs describe them.

5.1.1. TFI update

The firmware update can be done with a specific software tool provided by Telit that runs on Windows based PCs. The program will erase the content of flash memory, and then the program will write on the flash memory. “LE910_xxx_TFI.exe” file includes binary image.

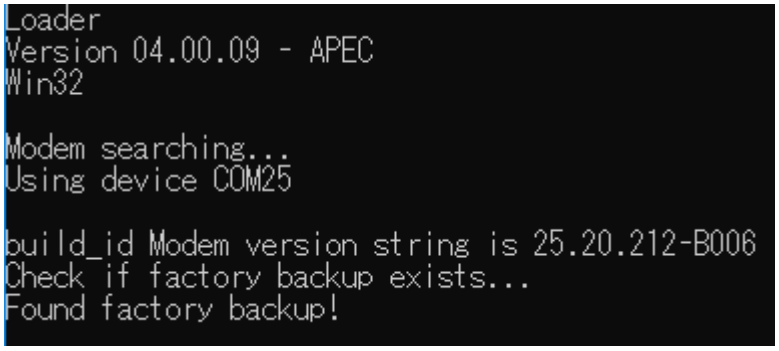
The following is procedure of TFI update.

1. Before update with TFI, please check “Telit Serial Diagnostics Interface” in your Windows device manager.



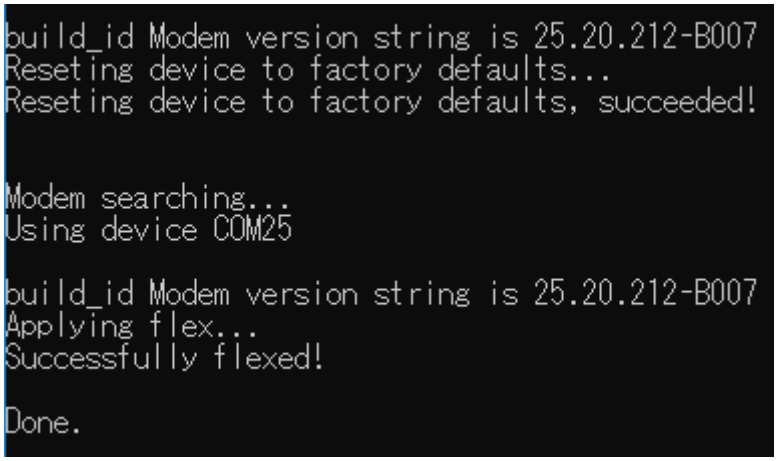
2. If you run LE910_xxx_TFI.exe, Windows CMD prompt will be popped up.

3. TFI downloader detects “Telit Serial Diagnostics” automatically, and start download.



4. Modem will reset several times during upgrade process.

5. Wait for the end of programming completed message



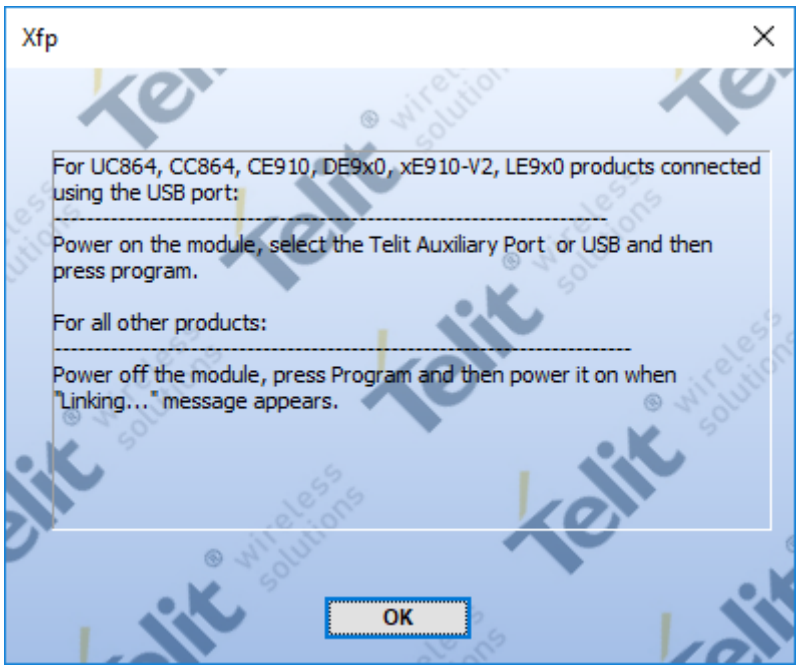
6. Telit LE910Cx module is now programmed with the new firmware.

5.1.2. XFP update

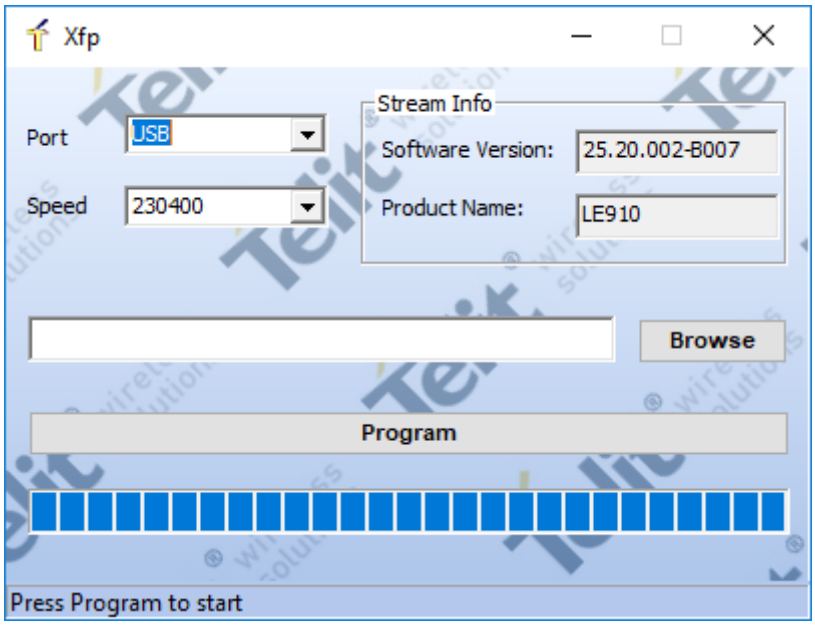
The firmware update of the module can be performed with the XFP Tool provided by Telit. It runs on Windows based PCs. It erases the flash memory content, and then it downloads the new firmware on the flash memory.

The following is procedure of TFI update.

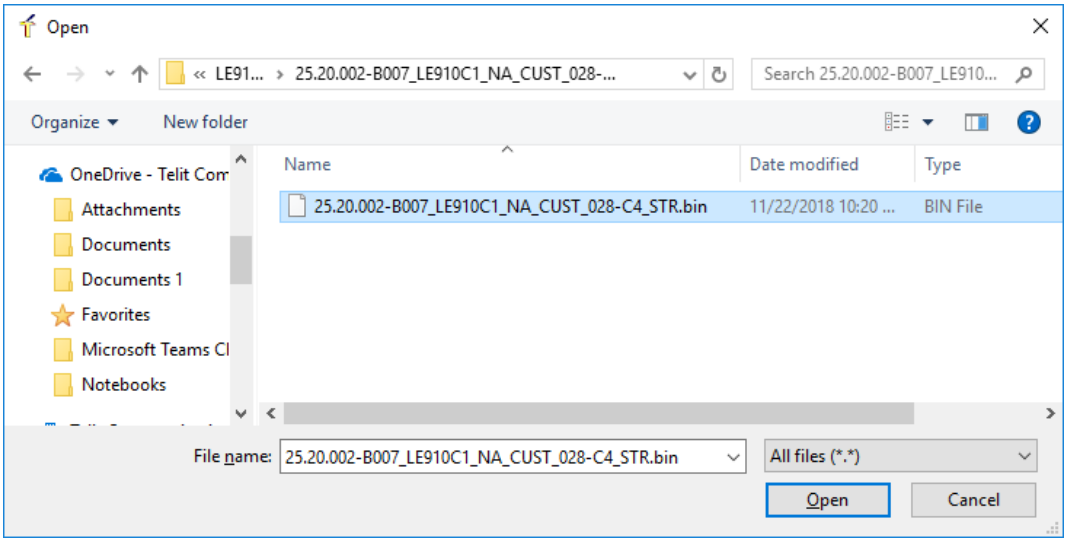
1. If you run “Xfp.exe”, the following window will be displayed. Please press ‘OK’ button after you finish to read the notice.



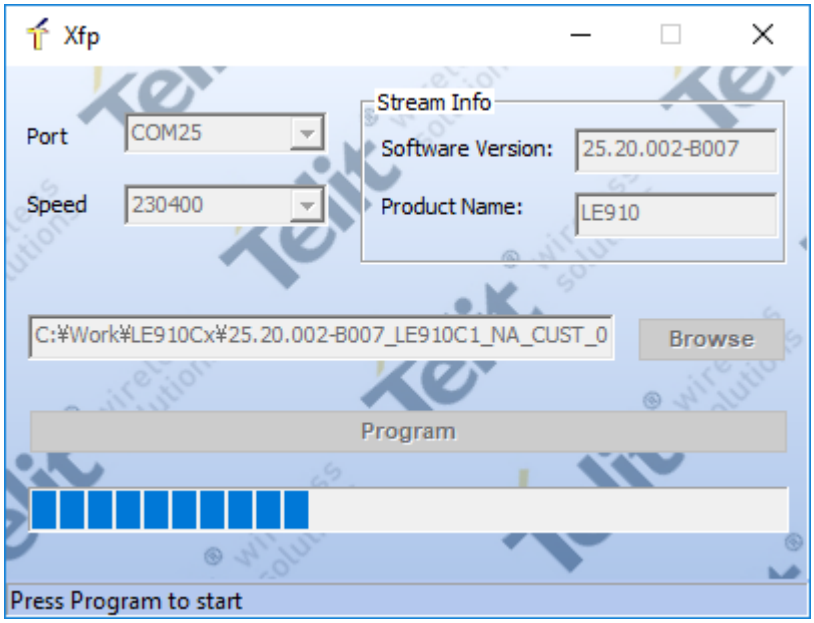
2. Please select “USB” in port combo box.



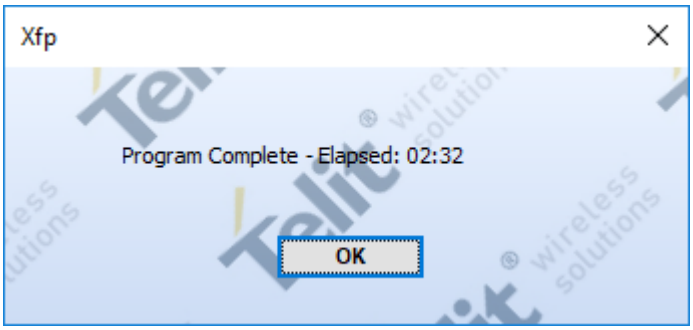
3. Press the ‘Browse’ button and select stream binary what you like to use for upgrade.



4. Press the ‘Program’ button for start upgrade. The blue bar will be increased during upgrade.



5. The following window is displayed on the screen when the module upgrade is success.



6. GLOSSARY AND ACRONYMS

APN	Access Point Name
BCCH	Broadcast Control Channel
CSD	Circuit Switched Data
CTM	Cellular Text Telephone Modems
CTS	Clear To Send
DCE	Data Circuit-Terminating Equipment (refer to [14])
DRX	Discontinuous Reception
DTE	Data Terminal Equipment (refer to [14])
DTMF	Dual Tone Multiple Frequency
DTR	Data Terminal Ready
GBR	Guaranteed Bit Rate
GERAN	GSM EDGE Radio Access Network
GPIO	General Purpose Input/Output
GUI	Graphic User Interface
HF	Hands Free (old terminology)
HS	Hand Set (old terminology)
HSPA	High Speed Packet Access
IMS	IP Multimedia Subsystem
IRA	International Reference Alphabet
ME	Mobile Equipment
MSISDN	Mobile Station International Subscriber Directory Number
NMEA	National Marine Electronics Association
NVM	Non-Volatile Memory
PDN	Public Data Network
PDP	Packet Data Protocol
PDU	Protocol Data Unit
PIN	Personal Identification Number
PPP	Point to Point Protocol
QoS	Quality of Service
SIM	Subscriber Identification Module

SMS	Short Message Service
SMSC	Short Message Service Center
TCP/IP	Transmission Control Protocol / Internet Protocol
TTY	Text Telephone Typewriter
UART	Universal Asynchronous Receiver Transmitter
UE	User Equipment
URC	Unsolicited Result Code
USIM	Universal Subscriber Identification Module
UTRAN	Universal Terrestrial Radio Access Network

7. DOCUMENT HISTORY

Revision	Date	Changes
1.0	2019-01-31	Initial version
2.0	2019-XX-XX	New: Update:

SUPPORT INQUIRIES

Link to **www.telit.com** and contact our technical support team for any questions related to technical issues.

www.telit.com

Telit Communications S.p.A.
Via Stazione di Prosecco, 5/B
I-34010 Sgonico (Trieste), Italy

Telit IoT Platforms LLC
5300 Broken Sound Blvd, Suite 150
Boca Raton, FL 33487, USA

Telit Wireless Solutions Inc.
3131 RDU Center Drive, Suite 135
Morrisville, NC 27560, USA

Telit Wireless Solutions Co., Ltd.
8th Fl., Shinyoung Securities Bld.
6, Gukjegeumyung-ro8-gil, Yeongdeungpo-gu
Seoul, 150-884, Korea

Telit Wireless Solutions Ltd.
10 Habarzel St.
Tel Aviv 69710, Israel

Telit Wireless Solutions
Technologia e Servicos Ltda
Avenida Paulista, 1776, Room 10.C
01310-921 São Paulo, Brazil