Linux SDK
Version 1.4.0.6
Release Notes

Document Revision:	1.0

Table of Contents

1 Introduction

This document contains technical details for the version 1.4.0.6 Linux SDK software release. It identifies the incremental changes to the SDK since its previous release. The previously released version of the SDK was 1.4.0.5.

Release 1.4.0.6 consists of bug fixes and code improvement/cleanup. Please review the Change Summary section in this document to help you become familiar with what's new in this release of SDK.

1.1 Change Summary

The table below contains hyperlinks to subsections of this document describing the changes.

Change Summary	
Item	**Title**
1	New Notification - SWI_NOTIFY_SmsDeliver – Receive GSM class 0/4 SMS
2	Mutex locking in logging
3	Update waiter to be independent of system clock
4	Doxygen Documentation Update

Additional details about these changes follow in the next subsections.

Throughout this document, if necessary, we refer to the root directory in which you have installed the SDK as

<SDKROOT>

1.1.1 New Notification - SWI_NOTIFY_SmsDeliver – Receive GSM class 0/4 SMS

This new notification enables application to receive GSM class 0/4 SMS via registered callback.
For an example of where this new notification is used, the reader is referred to:

Sample_Code/gsm/SwiSmsGsm/swismsgsm.c

1.1.2 Mutex locking in logging

Added mutex logging for dl module.

1.1.3 Update waiter to be independent of system clock

Replace timing logic from CLOCK_REALTIME to CLOCK_MONOTONIC

1.1.4 Doxygen Documentation Update

Updated Doxygen documentation for version 1.4.0.6.

1.2 Compatibility

1.2.1 Hardware Compatibility

There are two Sierra Wireless software components which have hardware dependencies built into them, the Sierra Wireless USB Driver and the SDK itself. This dependency is predominantly restricted to the USB Vendor ID and Product ID that the driver and SDK will detect, although there are additional dependencies in the driver as well, not discussed in this document.

1.2.2 Supported Architectures

This build of the Linux SDK contains object files for 32-bit Intel 80x86 architectures and for the recommended ARM 9 development platform. Sierra Wireless recommends the use of the Technologics TS-7800 single board computer for your early-stage development of ARM 9-based applications.

1.2.3 Supported Sierra Wireless Embedded Modems

The following Sierra Wireless modems are compatible with this version of the SDK. Note, not all the listed modems have been tested against this version of firmware:

Product Name
GSM
MC8775
MC8777
MC8704
MC8705
MC8780 / MC8781
MC8785
MC8790
MC8791/MC8792
MC8795v
MC8700
MC8801
SL8080
SL8081

SL8082
SL8083
SL8084
SL8085
SL8090
SL8091
SL8092
SL8093
CDMA
MC5725
MC5727
MC5727V
MC5728V
SL5010
SL5011
AC802

1.2.4 Software Compatibility

This version of the Linux SDK has been developed and tested using Ubuntu 13.04 LTS as installed from the Live CD *without* any subsequent upgrades applied. The as-delivered image contains source code, objects and executables for the supported i386/32 bit architecture.

1.2.4.1 C – Language

The Linux SDK is coded in the C-language and all external entry points (APIs) are callable by any language that can call C-Language functions. Some small changes to header files have been made to make calling C functions from the C++ language easier. The top and bottom of these header files have had the following code snippets added:

Top:

```
#ifdef __cplusplus
extern "C" {
#endif
```

Bottom:

```
#ifdef __cplusplus
}
#endif
```

1.2.4.2 Libraries

Customer applications can be built entirely from the library functions located in the *build/lib/<architecture>* directory.

Libraries are created using the Linux archive tool, “ar”, version 2.18.0.20080103

2 SDK Installation

This release of the SDK has been distributed as a tar'ed, zipped image. The steps for unpacking this software are described in detail in the following subsection.

2.1 Unpacking the Distributed Files

Use the following steps to unpack your Linux SDK release.

- Place the distribution file into a directory which will become the root directory for your Linux SDK directory tree.
- If you are using the command line, unzip the contents of the distributed file using the command:

  ```
  tar –xvvzf LinuxSDK_V1_4_0_6.tar.gz
  ```

- When this operation is complete, you will have a new directory with the same name as this Linux SDK distribution.

2.2 Verify the Contents

Executables for the Ubuntu 13.04 desktop environment are included with this distribution and should work out-of-the-box. A quick check to ensure the executables are able to run on your machine can be accomplished as follows:

- Open two shell windows and position them so they can be seen simultaneously. In one of the windows enter the following Linux command:

  ```
  tailf /var/log/user.log
  ```

  this will cause the contents of the user.log file to be printed to the console whenever any changes are made to that file.

- In the other shell window, ensure there is not already a previous version of the Linux SDK process running with the command:

  ```
  killall swisdk
  ```
  [1]

  This will cause an sdk process to be terminated immediately if there is one running.

- Change directory to

[1] Note, if you have been using older versions of the SDK you might have started the SDK daemon using its as-linked name, e.g. swisdki386. If there is a chance you have been running this executable then you must also make sure it is killed before attempting to use this new release.

```
LinuxSDK_V1_4_0_6/pkgs/ap
```

- Start this new version of the SDK by entering the following command into the same shell window you used to kill the currently-executing SDK (if any) in the previous step:

  ```
  ./aptesti386 –n t1 –p <SDKROOT>/build/bin/i386/swisdk
  ```

- If there is no Sierra Wireless modem installed on your computer, the test application will start and continuously display the following message until the SDK detects the presence of a modem:

  Air Server not available – SLEEPING

- You should be able to observe the log file (displayed in the other shell window) come alive with the display of several messages indicating the SDK process has started

If you have a compatible Sierra Wireless modem installed, after several seconds the aptesti386 command you entered above will terminate and the firmware version string from the attached modem will be displayed. The output will look similar to:

```
API Opened Successfully

Modem's boot and hold state is 0

Invoking test: t1, SwiGetFirmwareVersion

Test A: successful fetching of firmware version
Modem Firmware Version:
   J0_0_3_5AP C:/WS/FW/J0_0_3_5AP/MSM7200A/SRC/AMSS 2008/02/26 16:59:19

Test B: Empty firmware version buffer given
Buffer too small!

Test C: Small firmware version buffer given
Modem Firmware Version:
   J0_0_3_5AP


Test Completed ---------------------------------
```