| | |
|---|---|
| SIERRA WIRELESS | RTOS<br>2170021-2.0 |

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Author: | Sierra Wireless | | | | | | Date: | January 27, 2012 | | | |
| APN Content Level | BASIC | | INTERMEDIATE | ✓ | ADVANCED | | Confidentiality | Public | ✓ | Private | |
| Hardware Compatibility | Product Line | AirPrime | Series | Q26xx | | SL60xx | | | | | |
| | | | | WMPxx | | | | | | | |
| Software Compatibility | Series | ALL | | | | | | | | | |

# 1 Version

Application Notes may be updated over their lifetime. To ensure you design with the correct version, please check the application notes page in www.sierrawireless.com for latest versions.

# 2 Introduction

RTOS is an operating system which guarantees a certain capability within a specified time constraint. RTOS provides facilities which, if used properly, guarantees deadlines can be met generally (soft real-time) or deterministically (hard real-time).

This Application Note describes the general structure of the Real Time Operating System implemented in Sierra Wireless modules, the RTOS features extended to developers and their restrictions, limitations and best practices to be followed when using the RTOS features.

This Application Note (APN) is provided to Sierra Wireless distributors and clients to aid more rapid development of embedded applications using the Sierra Wireless portfolio of cellular solutions. To request a new application note, contact your regional Sierra Wireless Product Marketing Manager.

# 3 Overview

Sierra Wireless has a proprietary priority based, event driven soft RTOS. The RTOS is implemented in the firmware and its features are extended to the developers through Open AT Framework. Using Open AT Framework, a powerful embedded application environment, developers can create intelligent internet-enabled applications directly on the embedded module. However with earlier versions of Open AT Framework, it was not possible for developers to write time critical application along with GSM/GPRS capabilities. To overcome the limitation, Sierra Wireless has introduced the "RTOS" feature.

RTOS feature guarantees task execution within a specified time constraint. This feature allows user to:

- Access to various interrupts such as External Interrupts, BUS, Audio and Hardware Timers.
- Use of high granularity timers (<1ms).
- Access to low level interruption handlers.
- Access to high level interruption handlers.
- Multiple tasks and synchronization using semaphores.
- Configuration of processor clock speed.
- Predictable response time at each priority level.

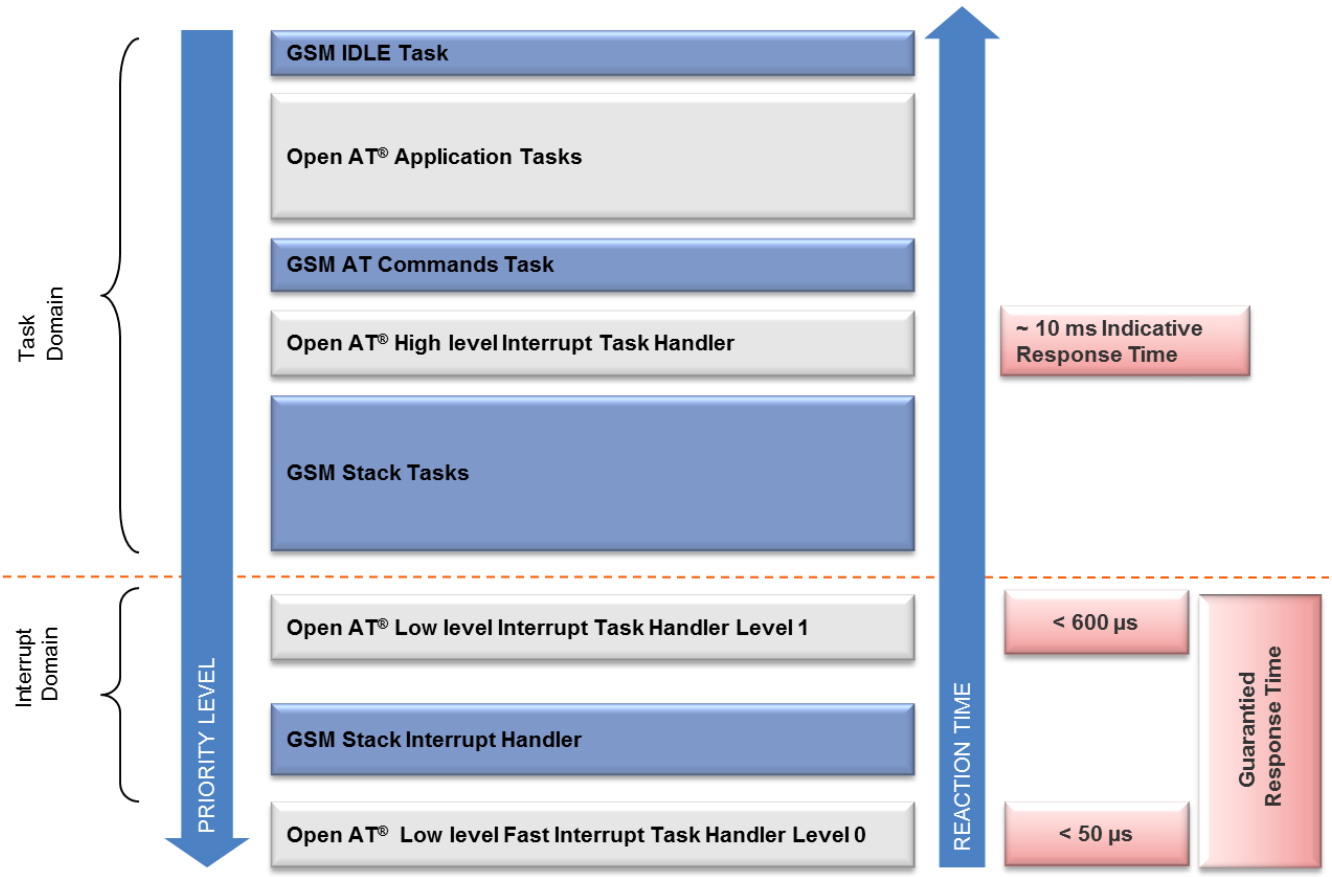
# 4 Glossary

| Initials | Definition |
|---|---|
| APN | Access Point Name |
| RTOS | Real Time Operating System |
| TCU | Timer and Capture Unit |
| RAM | Random Access Memory |

# 5 Sierra Wireless RTOS Architecture

Sierra Wireless RTOS has a fully pre-emptive priority based scheduling mechanism. This means that the firmware and the application are divided into multiple tasks, each having a different priority. Based on the priority, a task with higher priority can pre-empt a task with lower priority.

With the introduction of RTOS features to developers, Open AT Framework has given access to different priority levels to the application as described in the figure below.




The entire RTOS is broadly divided into two domains, viz. the Interrupt domain and the Task domain. Interrupt domain guarantees a specific response time based on the priority level, whereas the Task domain response time could vary depending on the processor load.

Using Open AT Framework OS, you can write your code in following areas:

- Low level Interrupt handlers
- High level interrupt handlers
- Applications

The Open AT Interrupt handler is the second highest (Level 1) or highest priority (Level 0) process on the platform. It is designed to perform asynchronous time critical operations. Its code would start being executed within 600 µs (Level 1) or 50 µs (Level 0) upon interruption (HW timer or external) detection. It is designed to perform very short operations such as value acquisition and storage in RAM. The Level 1 Interrupt handler has the highest priority than the Firmware tasks. This Interrupt handler can be attached to one of the following services:

- External interrupts
- Internal timer interrupts
- DSP (Audio) interrupts

The Open AT application can also be divided into multiple tasks, each having a different priority level defined by the user. All these tasks will sit in the Application Tasks area in the above figure, which means these priority levels will only divide the Application Tasks area into multiple tasks. These priority levels are not to be confused with other tasks (GSM, AT command, etc) priority levels.

There are different features of the RTOS that are exposed to the developers. These are discussed in brief in the following sections.

## 5.1 Multitasking

Multitasking allows a developer to have more than one task and up to 64 tasks in the application. Each task is assigned its own stack size, an entry point and a priority. When deciding the priority value, care should be taken to assign a different priority to each task to respect the scheduling policy of the RTOS.

With the availability of multiple tasks, there is a need to have some mechanism that allows managing, synchronizing and communicating between different tasks. The following sections explain the services that can be used with multitasking:

### 5.1.1 Task Management

With multiple tasks running, the application code is being executed from different tasks and interrupts. To manage these tasks and to get diagnostic information about various tasks, context service is provided. Using context service APIs, one can determine the current execution context, the states of the different tasks and the number of tasks in the application. It can also be used to perform some scheduling functions, for example to put a task to sleep or to suspend a task execution.

### 5.1.2 Inter Task Communication

Inter Task communication is possible using the message service. It allows every task to have a mailbox in the form of a call back function, and to send messages to other tasks. The task can also define a filter to be applied to incoming messages thereby optimizing the application performance.

### 5.1.3 Synchronization

In a multitasking environment, one often runs into problems where more than one task wants to access the same resource at the same time. Open AT Framework provides semaphores for synchronizing and restricting access to shared resources. Using semaphores, a part of the code can be declared as a critical section such that, when a lower priority task is executing that piece of code, no other task (even higher priority) can execute the same piece of code.

## 5.2 Interrupts

Interrupts are asynchronous breaks in program flow that occur as a result of events outside the running program. They are usually hardware related, occurring due to events such as a button press, timer expiration or completion of a data transfer. Through the Interrupt service, Open AT Framework allows to capture interrupts in the application using a call back function mechanism. The following services have to be used in order to capture interrupts in the application:

### 5.2.1 IRQ

IRQ is the basic framework that needs to be defined in the application in order to use the available interrupts. By defining the IRQ framework, the application gains access to the highest priority (as low level interrupt handler) level in the RTOS architecture. As the IRQ service gives access to the interrupt domain, it guarantees a response time based on the priority level of the interrupt handler (as described in the RTOS architecture above). Low level and high level interrupt handlers are defined using the IRQ service. Once defined, the interrupt handlers can be used with other services that require the IRQ framework, as described below.

### 5.2.2 External Interrupts

The embedded module has multiple dedicated external interrupts. These interrupts can be used in the application along with the IRQ framework. The priority of handlers will depend on the priority assigned while subscribing to the IRQ service.

### 5.2.3 Timer Interrupts

Sierra Wireless RTOS also provides access to accurate timers by means of the TCU service. Using the TCU service, the application can use high granularity accurate timers with a guaranteed response time, thanks to the IRQ framework. TCU service can also be used to monitor an external interrupt to capture and count the interrupts occurred during a defined time period.

### 5.2.4 Audio Interrupts

The IRQ framework also allows the application to have access to the DSP audio interrupts, thereby allowing playing or listening to a live audio stream. Whenever there is an incoming audio stream, either from the GSM network or from the microphone, the interrupt handler is called and the application has access to the incoming stream. Similarly, when the application wants to send an audio stream, the interrupt handler is called and the application can copy the data to be sent into the buffers.

# 6 Best Practices and Limitations

Since using the RTOS services the application can have the highest priority (even higher than the firmware and GSM interrupts), there are some key points that the application should take care of.

## 6.1 Low Level Handlers

1. It is important to optimize the code inserted in the low level handlers in order to minimize the impact in the multitasking environment. Only actions that have to be detected as soon as the interruption takes place should be monitored here. For example, copying some values in RAM buffers is a good practice, but performing complex, time consuming calculations should be avoided.

   For this reason, most of the APIs are not allowed to be called from the low level interrupt context and they return an error if done so.

2. Using an interrupt as FIQ can have serious impact on the embedded module behaviour including network registration, as the FIQ has the highest priority even above the GSM interrupts. Therefore, the code in the FIQ interrupt handler should be highly optimized.

3. A TCU timer is re-launched only when the corresponding low level handler function exits. Which means, in case of a cyclic timer of granuality x ms, the time taken for each call back would be x ms + the time taken by the low level handler to execute.

## 6.2 High Level Handlers

1. Though the High level handlers have a higher priority than the application task and are executed under the IRQ task context, the real time executing of these functions cannot be guaranteed. The response time of this function would vary based on the activity of the firmware tasks as these handlers have lower priority than the firmware tasks. Therefore, the application should not perform any real time critical operation in the high level handlers.

## 6.3 Multitasking

1. Some of the services provided should only be used from the highest priority task context. Whereas some of the service API calls are not allowed from the highest priority task context. The developer should understand the service description from the ADL user guide and it is always a good practice to monitor the return value of an API.

2. All the event call back functions except timers and message service will be executed in the highest priority task context (the main task context).

3. If a task is waiting for a Semaphore, it cannot be scheduled unless the corresponding semaphore is released by the other task. Hence it is important to produce a semaphore (after its consumption) to allow proper scheduling of other tasks.

4. Producing a semaphore too many times as compared to its consumption can result in a embedded module reset with an RTK exception 133.

## 6.4 RTOS and Sleep Mode

Sierra Wireless provides different low power consumption modes to reduce the power consumption based on the required functionalities. One of the widely used low power consumption modes is the Sleep Mode using the command AT+W32K. During the activation of this mode, the module uses a 32 KHz internal clock and can wake up on different events including some events in the application. The different events within the application that can wake up the module from the sleep mode are:

1. External interrupts.

2. Timer expiration, only if it is a strict timer.

Care should be taken when using the sleep mode along with RTOS features as it can impact the scheduling of the different application and interrupt tasks and TCU timers. The impact of Sleep Mode on different RTOS features is:

1. TasksWhen in sleep mode, tasks defined in the application and High level handlers will no longer be scheduled. The tasks will be executed the next time when the module wakes up from the sleep mode.

2. TCU timer expiry does not wake up the module from the Sleep Mode. Only an application timer defined as a strict timer can wake up the module. If a TCU timer expires when the module is in sleep mode, the corresponding handler will be invoked when the module comes out of the sleep mode.

3. External interrupts can be used to wake up the module from sleep mode.

# 7 Support

For direct clients: contact your Sierra Wireless FAE

For distributor clients: contact your distributor FAE

For distributors: contact your Sierra Wireless FAE

# 8 Document History

| Level | Date | History |
|---|---|---|
| 001 | January 22, 2009 | Creation |
| 002 | January 27, 2012 | New reference: 2170021<br>Old reference: WM_DEV_OAT_APN_017 |

# 9 Legal Notice

**Important Notice**

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

**Safety and Hazards**

Do not operate the Sierra Wireless modem in areas where blasting is in progress, where explosive atmospheres may be present, near medical equipment, near life support equipment, or any equipment which may be susceptible to any form of radio interference. In such areas, the Sierra Wireless modem **MUST BE POWERED OFF**. The Sierra Wireless modem can transmit signals that could interfere with this equipment. Do not operate the Sierra Wireless modem in any aircraft, whether the aircraft is on the ground or in flight. In aircraft, the Sierra Wireless modem **MUST BE POWERED OFF**. When operating, the Sierra Wireless modem can transmit signals that could interfere with various onboard systems.

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

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

**Limitations of Liability**

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

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

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

**Patents**

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

This product includes technology licensed from QUALCOMM®.

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

**Copyright**



**Trademarks**

AirCard® is a registered trademark of Sierra Wireless. Sierra Wireless™, AirPrime™, AirLink™, AirVantage™, Watcher™ and the Sierra Wireless logo are trademarks of Sierra Wireless.

wavecom®, , , ®, inSIM®, WAVECOM®, WISMO®, Wireless Microprocessor®, Wireless CPU®, Open AT® are filed or registered trademarks of Sierra Wireless S.A. in France and/or in other countries.

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

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

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

Other trademarks are the property of the respective owners.