
ZigBee® REva kit library package

Introduction

This document describes the ZigBee® REva Kit library package, which consists of two software libraries:

- EZSP library for driving the SN260 (by STMicroelectronics) or EM260 (by Ember) ZigBee devices.
- HAL library for addressing some of the REva platform devices and capabilities (buttons, buzzer, LEDs, memory, microcontroller, SPI protocol, system timer, UART). The HAL APIs can be matched with specific application purposes.

The EZSP APIs enable an ST microcontroller (hosted on the REva platform through a daughterboard) to communicate with the SN260 ZigBee network processor. This library is built on top of the HAL library, since each EZSP frame is sent serially to the SN260 silicon through the SPI interface (driven with the specific HAL APIs).

The Raisonance REva board, with a daughterboard supporting the selected ST microcontroller, and a radio communication module with the SN260 silicon are used as reference platforms.

This user guide is intended to provide an overall description of the software and hardware requirements for the ZigBee REva Kit library package, instructions for setting up the hardware and building the software library package as well as describing known limitations and issues at release time.

Due to the total compatibility between the SN260 and EM260, and the partnership between the two companies, some documents have been graciously supplied by Ember.

List of key words

- HAL: Hardware Abstraction Layer.
- EZSP: Ember ZigBee Serial Protocol.
- EmberZNet: ZigBee stack running over the SN260 silicon.
- RIDE: Raisonance Integrated Development Environment.
- RCM: Radio Communication Module.

Contents

- 1 Release notes 4**

- 2 Quick start 5**
 - 2.1 Required hardware 8
 - 2.1.1 External equipment 8
 - 2.2 Required software 8
 - 2.2.1 Host development platform (Ride7 toolset) 8
 - 2.2.2 Host development platform (IAR toolset) 9
 - 2.3 Hardware setup 9
 - 2.3.1 Module/cable connections 9
 - 2.3.2 REva board jumper settings 10
 - 2.3.3 REva Power area 11
 - 2.3.4 Daughterboard jumper settings 11
 - 2.3.5 Raisonance RLink jumper settings 13
 - 2.4 Building the library 13

- 3 Application examples 14**
 - 3.1 Setup the application serial communication channel 14
 - 3.2 Version application 15
 - 3.2.1 To run the version application 16
 - 3.2.2 Load and run a version pre-built image 16
 - 3.2.3 Building/running steps for STM32F103x and STR71x-STR91x microcontrollers using IAR toolset 17
 - 3.3 Sensor, sleepy sensor and sink applications 17
 - 3.3.1 To run the sensor application 18
 - 3.3.2 Load and run three sensor, sleepy sensor and sink pre-built images .. 18
 - 3.3.3 Building/running steps for STM32F103x and STR71x-STR91x microcontrollers using IAR toolset 19
 - 3.4 Light and switch applications 20
 - 3.4.1 To run the light application 20
 - 3.4.2 Load and run two light and switch pre-built images 21
 - 3.4.3 Building/running steps for STM32F103x and STR71x-STR91x microcontrollers using IAR toolset 22

- 4 SPI bootloader 23**

5 Limitations and support 23

6 Revision history 24

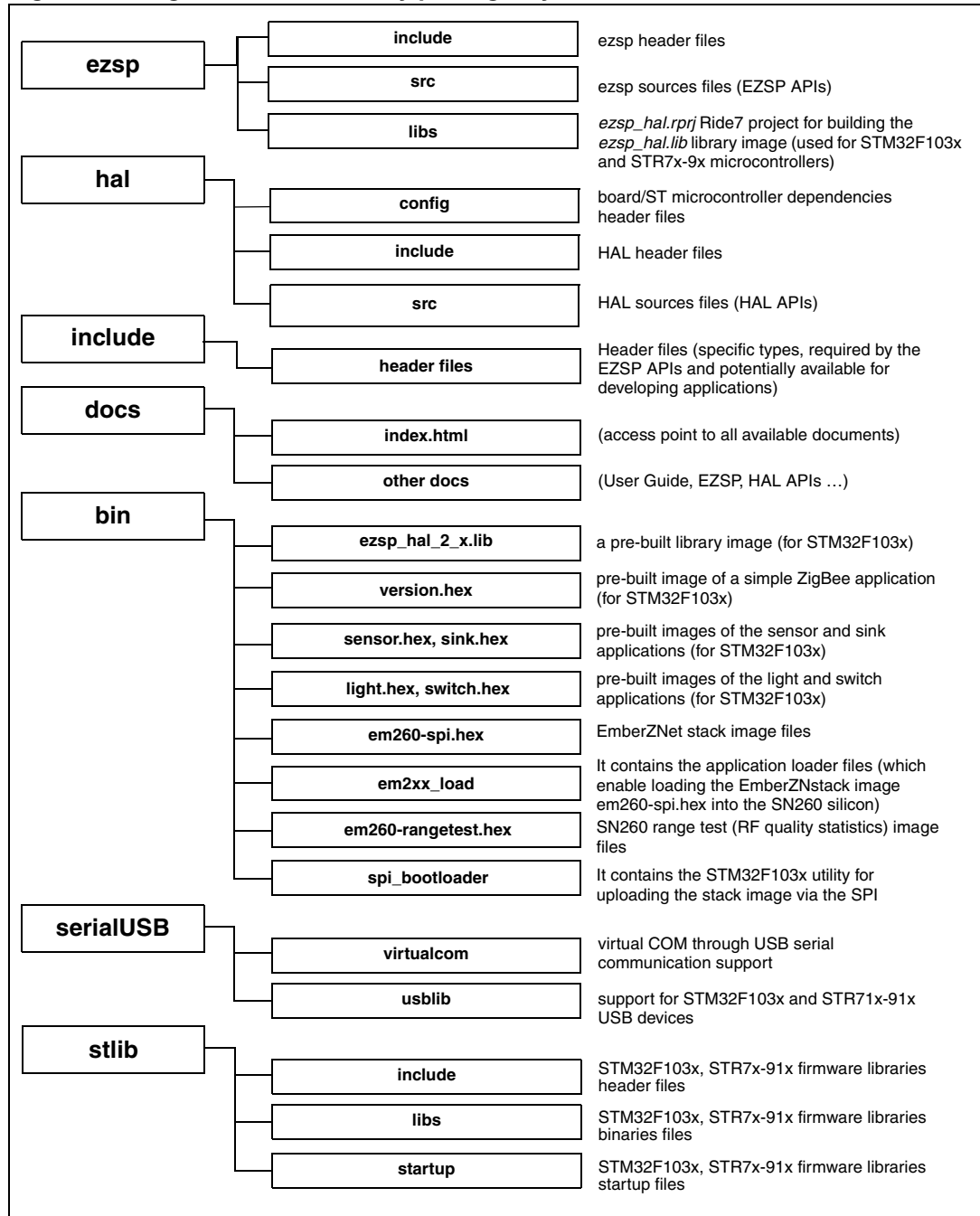
1 Release notes

The current release supports the STM32F103x, STR71xF, STR75xF and STR91xF microcontrollers.

2 Quick start

Figure 1 describes the ZigBee REva Kit library package layout.

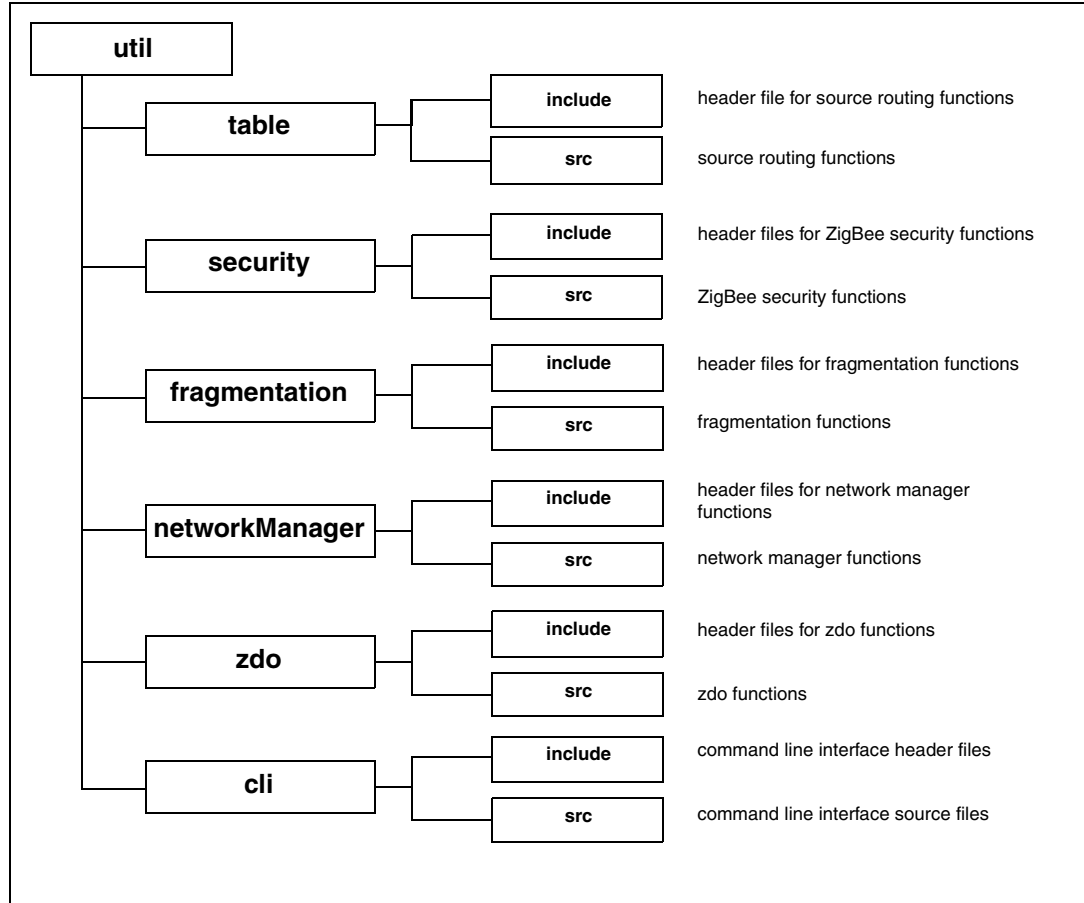
Figure 1. ZigBee REva Kit library package layout



The directory *util* includes functions for handling the ZigBee source routing, security, fragmentation, network, ZDO and command line interface.

Figure 2 describes the *util* directory layout.

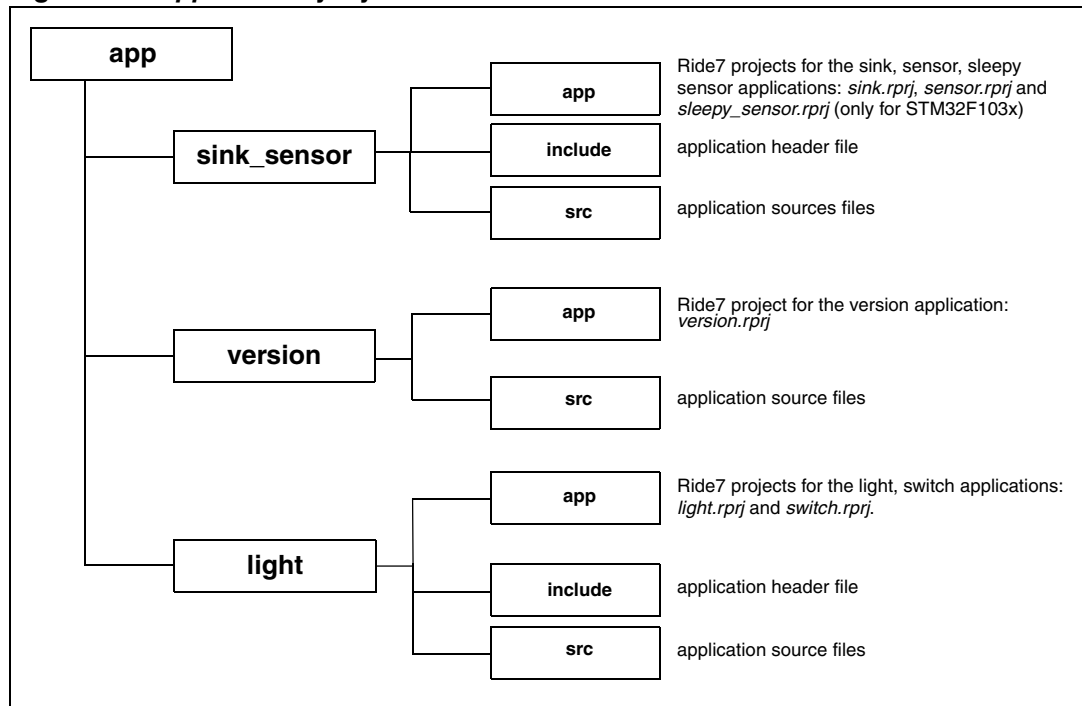
Figure 2. *util* directory layout



In addition, a directory *app* is provided for hosting applications built over the library. Some applications are provided for showing the library package functions: a simple application which enables basic ZigBee operations to be performed, two simple applications (light and switch) for controlling a light and three more complex applications (sink, sensor, sleepy sensor) which enable the configuration of a distributed sensors network.

Figure 3 describes the *app* directory layout.

Figure 3. *app* directory layout



Furthermore, for each application, the corresponding IAR workspace is provided to enable the user to directly build the application using the IAR toolset.

- Note:**
- 1 The *ezsp_hal* library does not provide an IAR workspace.
 - 2 This version of the ZigBee® REva Kit package provides the IAR workspaces only for STM32F103x, STR71x and STR91x microcontrollers.
 - 3 The sleepy sensor application is only supported by the STM32F103RBT6 microcontroller.
 - 4 The available pre-built images of *version.hex*, *light.hex*, *switch.hex*, *sensor.hex*, *sink.hex*, *ezsp_hal.lib* are provided only for STM32F103x microcontrollers (not the STR7x or STR91x).

2.1 Required hardware

The target hardware is described below.

- REva ZigBee Starter Kit for ST Microcontrollers which includes:
 - Raisonance REva board v2.12
 - RLink v.2.10
 - REva daughterboard (with the ST microcontroller)
 - SN260 radio communication module
 - REva USB cable for programming the ST microcontroller with the RIDE toolset (and also to power the overall board)
- Cabling:
 - Serial cable (for connecting the REva board serial SER1 to a PC RS232 port)
 - USB mini cable (for connecting the STM32F103x and STR71x-9x USB mini-B connector to a PC USB port)

Note: At the time of manufacture, the REva board is provided in the “Attached mode”: REva and RLink boards are connected, no connection cable is required and power is supplied by the USB connection. This is the standard configuration used for the library, applications building and running.

Refer to the REva documentation for information corresponding to the REva + RLink + ST daughterboards.

The EM260 device (Ember vendor) and the SN260 (STMicroelectronics vendor) are totally compatible.

2.1.1 External equipment

To show the library functions, the minimum external equipment required is a PC (Windows® XP or 2000) with the Raisonance Ride7 toolset (see software section for more information).

When using the IAR toolset, the following equipment is required for programming (and/or debugging):

- IAR J-Link JTAG emulator hardware for programming/debugging
- External power supply (9 V) for powering the REva board

2.2 Required software

The library is distributed through a specific link on the STMicroelectronics web site.

2.2.1 Host development platform (Ride7 toolset)

The Raisonance Ride7 toolset (Ride7 IDE version 7.02.001 and RKit ARM for Ride7 version 1.05.001) is used for building libraries and applications running with STM32F103x and STR7x-91x ARM microcontroller families. Ride7 is the new integrated development environment designed for the development of STMicrocontroller projects. This tool automatically manages file dependencies so that no makefile is necessary.

The toolset is available from www.stm32circle.com/resources/tools.php. Select “Download the latest CD-ROM image”.

2.2.2 Host development platform (IAR toolset)

The IAR Embedded Workbench IDE for ARM toolset (version 4.42A supports the ARM Cortex™) is also used for building and running applications. The IAR Embedded Workbench IDE for ARM is a very powerful integrated Development Environment designed for developing and managing complete embedded applications projects.

- Note:*
- 1 For programming (and/or debugging) using the IAR toolset, an IAR J-Link JTAG emulator is required. The IAR J-Link is a JTAG emulator designed for ARM cores. It connects via USB to a PC running Windows 2000 or XP. It has a built-in 20-pin JTAG connector, which is compatible with the standard 20-pin connector defined by ARM. Using the IAR J-Link, an external power supply is required for powering the REva board. The 9V power supply should output 9V DC and have a 2.1 x 5 mm jack with the ground signal on the outside.
 - 2 The first time the IAR J-link is plugged in the PC USB port, the user is requested to provide the corresponding J-Link driver, browsing to the IAR installation directory and selecting the folder ARM, drivers, Jlink.
 - 3 When disconnecting the IAR J-Link from the PC USB port, also unplug the IAR 20-pin JTAG connector from the REva on-board 20-pin JTAG ISD connector.
 - 4 The IAR toolset and the IAR J-Link are not provided with the kit. For detailed information about the IAR products refers to the www.iar.com web site.

2.3 Hardware setup

2.3.1 Module/cable connections

Please ensure that the target hardware is connected as described below:

- 1 Plug the SN260 RCM module on the REva Raisonance two 6-pins, single row, and 0.1-inch pitch sockets present in the wrapping zone (they enable direct connection of the RCM module to the REva ZigBee platform).
- 2 Plug the ST microcontroller daughterboard into the specific REva socket.

When using the Ride7 toolset:

- 3 Set the REva power jumper according to the settings described in [Section 2.3.3 on page 11](#) and plug the REva USB cable into the PC USB port and the RLink port.

When using the IAR toolset:

- 4 Set the REva power jumper according to the settings described in [Section 2.3.3 on page 11](#) and plug an external power supply connector into the REva 9V DC input jack.
- 5 Plug the IAR J-Link 20-pin JTAG connector to the REva on-board 20-pins ISD connector and the J-Link USB cable to a PC USB port.

If the application requires displaying debugging messages and/or interaction with the user, the following serial communication channels are supported:

- Serial COM through RS232 (for all microcontrollers):
 - Connect a serial cable between the PC serial port and the REva SER1 port.
- Virtual COM through USB (only for STM32F103x and STR71x-91x microcontrollers):
 - Connect the USB-Mini cable to the STM32F103x and STR71x-9x USB mini-B connector and to a PC USB port.

Caution: When unplugging the USB cable from the mini-B connector of the STM32F103x and STR71x-9x daughterboards, always hold the daughterboard with one hand while removing the cable with the other. Otherwise, the daughterboard could be torn off from the SO-DIMM connector.

2.3.2 REva board jumper settings

Table 1 lists the jumper settings for REva board functional areas (inputs, outputs, analog, com with the STM32F103x, STR71x, STR75x and STR91x daughterboards).

Table 1. Settings for STM32F103x, STR71x, STR75x and STR91x daughterboards

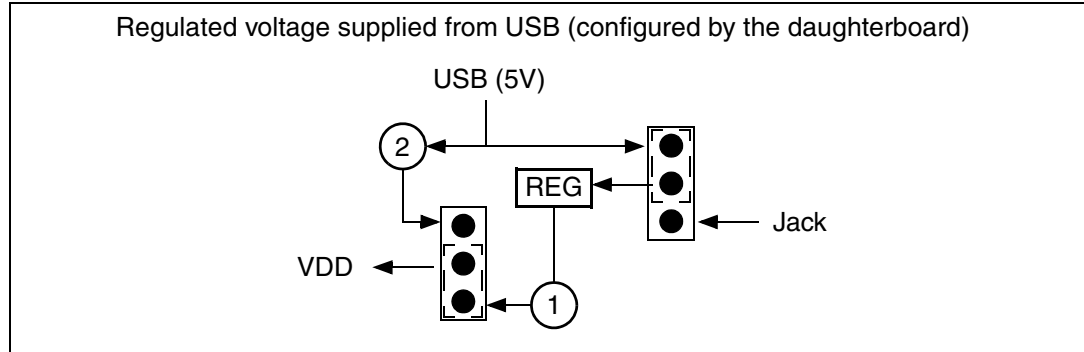
Jumper	Setting	Purpose
D7	Fitted	Enable LED D7 (available for application use).
D6	Fitted	Enable LED D6 (available for application use).
D5	Fitted	Enable LED D5 (available for application use).
D4	Unfitted	Exclusively used for the SN260 to the ST microcontroller serial communication over the SPI interface.
D3	Unfitted	Exclusively used for the SN260 to the ST microcontroller serial communication over the SPI interface.
D2	Fitted	Enable LED D2 (available for application use).
D1	Unfitted	Exclusively used for the SN260 to the ST microcontroller serial communication over the SPI interface.
D0	Unfitted	Exclusively used for the SN260 to the ST microcontroller serial communication over the SPI interface.
BT5	Fitted	Enable the BT5 button (available for application use).
BT6	Fitted	Enable the BT6 button (available for application use). Note that the POL jumper also has to be fitted to _ _ position.
POL	Fitted as _ for STM32F103x and STR91x. Fitted as _ _ for STR71x, STR75x	Enable the BT6 button usage.
BUZZ	Fitted	Enable buzzer (available for application use).
TX	Fitted	Enable serial transmission (available for application use).
RX	Fitted	Enable serial reception (available for application use).

2.3.3 REva Power area

Ride7 environment

When using the REva power jumper settings in a Ride7 environment, keep the default configuration (USB power supply) as shown in [Figure 4](#).

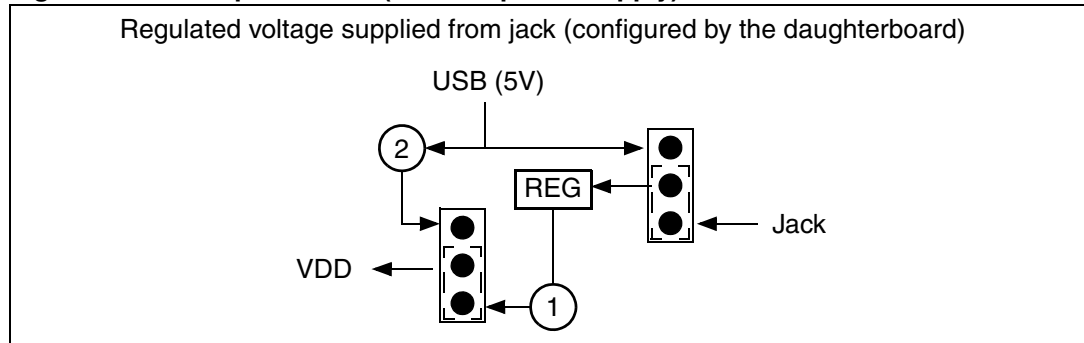
Figure 4. REva power area (USB power supply)



IAR environment (IAR J-Link JTAG emulator and external power supply)

When using the REva power jumper settings in an IAR environment, set the jumper for an external power supply as shown in [Figure 5](#).

Figure 5. REva power area (external power supply)



2.3.4 Daughterboard jumper settings

REva STM32F103x daughterboard

For the REva STM32F103x daughterboard, [Table 2](#) and [Figure 6](#) show the default settings.

Table 2. REva STM32F103x daughterboard

Jumper	Setting	Purpose
FLASH	Fitted	Flash boot mode
RAM	Unfitted	RAM boot mode (the Flash boot mode is used)

Figure 6. Boot mode setting for STM32F103x daughterboard



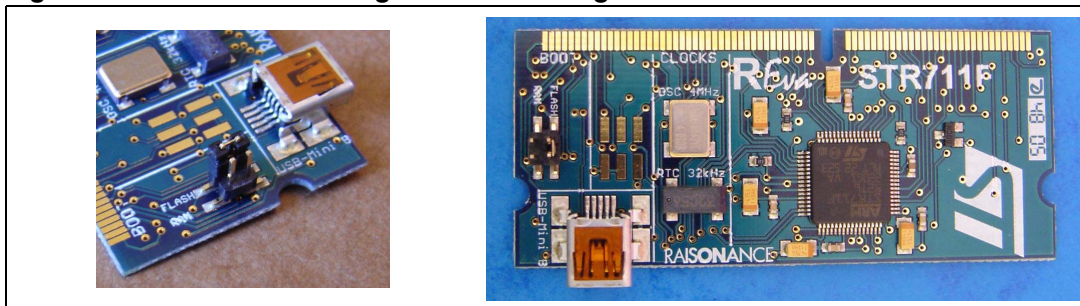
REva STR71x daughterboard

For the REva STR71x daughterboard, [Table 3](#) and [Figure 7](#) show the default settings.

Table 3. REva STR71x daughterboard

Jumper	Setting	Purpose
FLASH	Fitted	Flash boot mode
RAM	Unfitted	RAM boot mode (the Flash boot mode is used)

Figure 7. Boot mode setting for STR71x daughterboard



REva STR75x daughterboard

For the REva STR75x daughterboard, [Table 4](#) shows the default settings.

Table 4. REva STR75x daughterboard

Jumper	Setting	Purpose
Boot0 = 0	Fitted	Flash boot mode
Boot0 = 1	Unfitted	Flash boot mode
Boot1 = 0	Fitted	Flash boot mode
Boot1 = 1	Unfitted	Flash boot mode

Note: The STR91x cannot boot from RAM. As a consequence, no specific boot jumpers are present on the STR91x daughterboard.

2.3.5 Raisonance RLink jumper settings

When using STM32F103x, STR71x, STR75x and STR91x daughterboards, just keep the default jumper settings as shown in [Figure 8](#).

Figure 8. Power area for Raisonance RLink (STM32F103x, STR71x, STR75x and STR91x microcontrollers)



2.4 Building the library

The library software relies on the software and build system framework introduced in the library layout section.

What follows is a short introduction, aimed at providing enough knowledge to be able to build the software library using the Ride7 toolset.

STM32F103x, STR7x-9x microcontrollers specific library building steps (Ride7 toolset)

1. Open the Ride7 toolset.
2. From the **Project, Open Project** menu, open the `ezsp\libs\ezsp_hal.rprj` Ride7 project.
3. From the **Options, Project Properties, Configuration** menu, select **ST_STM32_2_x**.
4. From the **Project** menu, select **Build Project**: a library `ezsp_hal_2_x.lib` is produced in the `ezsp\libs` directory.

- Note:**
- 1 The `ST_STM32_2_x` configuration builds the `ezsp_hal` library image using the latest available STM32F10xxx V2.0 firmware library. The `ST_STM32_1_x` configuration is used to build the `ezsp_hal` library image (`ezsp_hal_1_x.lib`) using the STM32F10xxx V1.0 firmware library.
 - 2 For building the `ezsp_hal` library image for STR711FR2, STR912FW44 and STR750FV2 microcontrollers, select the related configurations from the **Options, Project Properties, Configuration** menu.

3 Application examples

Some applications are provided for showing the library package functions.

3.1 Setup the application serial communication channel

The provided application examples require displaying debugging messages and/or interaction with the user. Two serial communication channels are supported: serial COM through RS232 or virtual COM through USB.

Setup a serial COM through RS232 (for all microcontrollers)

1. Connect a serial cable between the PC serial port and the REva SER1 port.
2. Open a HyperTerminal on the serial COM port with the following configuration (the application messages and/or interactions come through the serial HyperTerminal):
 - Bit rate: 9600
 - Data bits: 8
 - Parity: None
 - Stop bits: 1
 - Flow control: None

Setup a virtual COM communication through USB (only for STM32F103x and STR71x-91x microcontrollers)

1. Connect the USB-Mini cable to the STM32F103x or STR71x-91x USB mini-B connector and to a PC USB port. (The first time a STM32F103x or STR71x-91x USB device is plugged to the PC USB port, the user is required to install the corresponding USB software driver; select file **stmcdc.inf** in the **serialUSB** directory).
2. Using the mouse, right-click on My Computer, select **Manage, Device Manager**, and open **Ports (COM & LPT)** to display a STM32F103x or STR71x-91x CDC communication port on a specific COMx port. The STM32F103x or STR71x-91x USB device has been recognized.
3. Open a HyperTerminal on the corresponding USB virtual COMx port with the following configuration:
 - Bit rate: 9600
 - Data bits: 8
 - Parity: None
 - Stop bits: None
 - Flow control: None

Reset a virtual COMx communication channel

1. Disconnect the COMx HyperTerminal.
2. Reset the REva board (STM32F103x or STR91x case) or Power OFF/ON the board (STR71x case).
3. Make a call on the COMx HyperTerminal. The application messages and/or interactions come through the COMx HyperTerminal.

- Note:
- 1 When resetting the STR711FR2 microcontroller (using the REva RESET button), the PC is not able to enumerate again the STR71x USB device. To reset the STR71x USB device, power OFF/ON the REva board.
 - 2 The PC is able to recognize only a single STR71x-91x USB device when plugged on a PC USB port. When connecting a second STR71x-91x USB device to another PC USB port, the PC will not recognize it.
 - 3 If both the connectors (USB-mini and RS232) are connected, the virtual COM through USB communication is automatically selected by the application.

3.2 Version application

The version application is a simple application used to perform basic ZigBee operations:

1. Initialize the EmberZnet Stack.
2. ezspVersion for getting the EmberZnet stack version running on the SN260 silicon.
3. Get the Eui64 node address.

The following paragraphs provide a short introduction about how to build and run the application.

Specific building steps for STM32F103x and STR7x-9x microcontrollers (Ride7 toolset)

1. Open the Ride7 toolset and open the Ride7 *ezsp_hal.rprj* project.
2. Compile the library following the instructions in [Section 2.4 on page 13](#).
3. From the **Project, Open Project** menu, open the *app\version\app\version.rprj* Ride7 project.
4. From the **Options, Project Properties, Configuration** menu, select **ST_STM32_2_x**.
5. From the **Project** menu, select **Build Project**: a binary file version is produced in the *app\version\app* directory.

- Note:
- 1 The *ST_STM32_2_x* configuration builds the application image using the latest available STM32F10xxx V2.0 firmware library. The *ST_STM32_1_x* configuration is used to build the application image using the STM32F10xxx V1.0 firmware library.
 - 2 For building the application image for the STR711FR2, STR912FW44 and STR750FV2 microcontrollers, select the related configurations from the **Options, Project Properties, Configuration** menu.

3.2.1 To run the version application

1. Connect the REva USB cable between the PC and the RLink.

Running steps for STM32F103x and STR7x-9x microcontrollers (Ride7 toolset)

2. From the **Options, Project Properties, Cortex RLink, Advanced Options** menu, open **Advanced Options** and select **Debug** for STM32F103x microcontrollers or unselect **Debug** for STR7x-STR9x microcontrollers.
3. From the **Debug** menu, select **Start**. The version application image is then downloaded.
4. Using the STM32F103x microcontroller, from the **Debug** menu, select **Run** and then **Terminate**.
5. Set up the serial communication channel as described in [Section 3.1 on page 14](#).
If using the virtual COM, reset the corresponding communication channel as described in [Section 3.1 on page 14](#).

After setting the serial communication channel, certain messages are displayed in the PC HyperTerminal window. The example below shows a log of a version application execution:

```
***** Stack initialization performed with success
***** Stack Version = 3325
***** EUI64 Node ID = 000D6F000009A340.
```

Note: The most recent EmberZNet 3.x stack version is 3325.

3.2.2 Load and run a version pre-built image

A pre-built image of the version application for the STM32F103x microcontroller is already present in the *bin* directory. To download the *version* image into the ST microcontroller Flash and run the version application, follow these steps:

1. Connect the REva USB cable between the PC and the RLink.

Steps for STM32F103x and STR7x-9x microcontrollers (Ride7 toolset)

2. Open the Ride7 toolset.
3. From the **Options, Project Properties, Advanced ARM Options, Processor** menu, open **Device** and select the specific STM32F103x or STR7x-9x microcontroller (STM32F103RBT6 as STM32F103x microcontroller, STR711FR2 as STR71x microcontroller, STR750FV2 as STR75x microcontroller, or STR912FW44 as STR91x microcontroller).
4. From the **Options, Project Properties, Advanced ARM Options, Debug environment** menu, open **Debug tool** and select **RLink**.
5. From the **Options, Project Properties, Cortex RLink, Advanced Options** menu, open **Advanced Options** and deselect **Debug**.
6. From the **Options, Project Properties, Cortex RLink, Advanced Options** menu, open **Advanced Options** and select **Erase target now!**
7. From the **Options, Project Properties, Cortex RLink, Advanced Options** menu, open **Advanced Options**, select **Write Target Flash Now** and select the *bin\version.hex* image file. Wait for the image to download into the Flash memory.
8. Set up the serial communication channel as described in [Section 3.1 on page 14](#).
9. If using the virtual COM, reset the corresponding communication channel as described in [Section 3.1 on page 14](#).

3.2.3 Building/running steps for STM32F103x and STR71x-STR91x microcontrollers using IAR toolset

1. Open the IAR toolset.
2. From the **File, Open, Workspace** menu, open the `app\version\app\IAR\version.eww` IAR workspace.
3. From the **View** menu, select **Workspace** to display supported projects. (A window opens on the left side of the IAR environment.)
4. From the **Workspace** selector window, select the application configuration according to the microcontroller to be addressed:
 - `version_STM32_2_x` configuration for the STM32F103x microcontroller with firmware library V2.0
 - `version_STM32_1_x` configuration for the STM32F103x microcontroller with firmware library V1.0
 - `version_STR71x` configuration for the STR71x microcontroller
 - `version_STR91x` configuration for the STR91x microcontroller
5. From the **Project** menu, select **Rebuild All**. An Intel HEX file is built in the `app\version\app\IAR` directory.

To download the version application

1. From the **Project** menu, select **Debug**. Wait for the image to download into the Flash memory.
2. From the **Debug** menu, select **GO**.
3. From the **Debug** menu, select **Stop Debugging**.
4. Set the serial communication channel as described in [Section 3.1 on page 14](#).
5. If using the virtual COM, reset the corresponding communication channel as described in [Section 3.1 on page 14](#).

3.3 Sensor, sleepy sensor and sink applications

The sensor, sleepy sensor and sink applications set a distributed sensors network. To build and run the sensor application, follow these steps:

Building steps for STM32F103x and STR7x-9x microcontrollers (Ride7 toolset)

1. Open the Ride7 toolset and open the Ride7 `ezsp_hal.rprj` project.
2. Compile the library following the instructions in [Section 2.4 on page 13](#).
3. From the **Project, Open Project** menu, open the `app\sink_sensor\app\sensor.rprj` Ride7 project.
4. From the **Options, Project Properties, Configuration** menu, select **ST_STM32_2_x**.
5. From the **Project** menu, select **Build Project**: a binary file sensor is produced in the `app\sink_sensor\app` directory.

- Note:*
- 1 The `ST_STM32_2_x` configuration builds the application image using the latest available STM32F10xxx V2.0 firmware library. The `ST_STM32_1_x` configuration is used to build the application image using the STM32F10xxx V1.0 firmware library.
 - 2 For building the application image for the STR711FR2, STR912FW44 and STR750FV2 microcontrollers, select the related configurations from **Options, Project Properties, Configuration** menu.

3.3.1 To run the sensor application

1. Connect the REva USB cable between the PC and the RLink.

Running steps for STM32F103x and STR7x-9x microcontrollers (Ride7 toolset)

2. From the **Options, Project Properties, Cortex RLink, Advanced Options** menu, open **Advanced Options** and select **Debug** for STM32F103x microcontrollers and unselect **Debug** for STR7x-STR9x microcontrollers.
3. From the **Debug** menu, select **Start**. The sensor application image is then downloaded.
4. Using the STM32F103x microcontroller, from the **Debug** menu, select **Run** and then **Terminate**.
5. Set up the serial communication channel as described in [Section 3.1 on page 14](#).
6. If using the virtual COM, reset the corresponding communication channel as described in [Section 3.1 on page 14](#).
7. Follow the messages displayed in the HyperTerminal window for interacting with the sensor node and to see how this node communicates with a sink node. (The sensor node expects to send data to a sink node.)

Note: For building and running the sink and sleepy sensor applications, just open the `app\sink_sensor\app\sink.rprj` and `app\sink_sensor\app\sleepy_sensor.rprj` Ride7 projects. Follow the same steps as for the sensor application, adding the `SINK_APP` (`SLEEPY_SENSOR_APP`) define value instead of the `SENSOR_APP` one. Furthermore, another PC serial communication channel (serial RS232 or virtual COM through USB) is required for user interaction with the sink and sleepy sensor applications.

The sleepy sensor application does not work using the USB virtual COM communication channel (MBTst33602).

For a description of the sink and sensor applications, refer to the related documentation.

3.3.2 Load and run three sensor, sleepy sensor and sink pre-built images

Three pre-built-images of the subscribed sensor, sleepy sensor and sink applications for the STM32F103x microcontroller are already present in the `bin` directory. To download the `sensor.hex` image into the ST microcontroller Flash and run the sensor application, follow these steps:

1. Connect the REva USB cable between the PC and the RLink.

Steps for STM32F103x and STR7x-9x microcontrollers

2. Open the Ride7 toolset.
3. From the **Options, Project Properties, Advanced ARM Options, Processor** menu, open **Device** and select the specific STM32F103x or STR7x-9x microcontroller (STM32F103RBT6 as STM32F103x microcontroller, STR711FR2 as STR71x microcontroller, STR750FV2 as STR75x microcontroller, or STR912FW44 as STR91x microcontroller).
4. From the **Options, Project Properties, Advanced ARM Options, Debug environment** menu, open **Debug tool** and select **RLink**.
5. From the **Options, Project Properties, Cortex RLink, Advanced Options** menu, open **Advanced Options** and deselect **Debug**.

6. From the **Options, Project Properties, Cortex RLink, Advanced Options** menu, open **Advanced Options** and select **Erase target now!**
7. From the **Options, Project Properties, Cortex RLink, Advanced Options** menu, open **Advanced Options**, select **Write Target Flash Now** and select the *bin\sensor.hex* image file. Wait for the image to download into the Flash memory.
8. Set up the serial communication channel as described in [Section 3.1 on page 14](#).
9. If using the virtual COM, reset the corresponding communication channel as described in [Section 3.1 on page 14](#).

Follow the same steps for configuring another REva board as a sink or sleepy sensor node (using the *bin\sink.hex* or *bin\sleepy_sensor.hex* pre-built images). Additionally, another PC serial communication channel (serial RS232 or virtual COM through USB) is required for user interaction with the sink or sleepy sensor node.

Note: The sleepy sensor application does not work using the USB virtual COM communication channel (MBTst33602).

3.3.3 Building/running steps for STM32F103x and STR71x-STR91x microcontrollers using IAR toolset

1. Open the IAR toolset.
2. From the **File, Open, Workspace** menu, open the *app\sink_sensor\app\IAR\sensor.eww* IAR workspace.
3. From the **View** menu, select **Workspace** to display supported projects. (A window appears on the left side of the IAR environment.)
4. From the **Workspace** selector window, select the application configuration according to the microcontroller to be addressed:
 - sensor_STM32_2_x configuration for the STM32F103x microcontroller with firmware library V2.0
 - sensor_STM32_1_x configuration for the STM32F103x microcontroller with firmware library V1.0
 - sensor_STR71x configuration for the STR71x microcontroller
 - sensor_STR91x configuration for the STR91x microcontroller
5. From the **Project** menu, select **Rebuild All**. An Intel HEX file is built in the *app\sink_sensor\app\IAR* directory.

To download the sensor application

1. From the **Project** menu, select **Debug**. Wait for the image to download into the Flash memory.
2. From the **Debug** menu, select **GO**.
3. From the **Debug** menu, select **Stop Debugging**.
4. Set up the serial communication channel as described in [Section 3.1 on page 14](#).
5. If using the virtual COM, reset the corresponding communication channel as described in [Section 3.1 on page 14](#).

When building and running the sink or sleepy sensor application, just open the *app\sink_sensor\app\IAR\sink.eww* or *app\sink_sensor\app\IAR\sleepy_sensor.eww* IAR workspace and follow the same steps as for the sensor application.

3.4 Light and switch applications

The light and switch applications control the switching on/off of a light.

Building steps for STM32F103x and STR7x-9x microcontrollers (Ride7 toolset)

1. Open the Ride7 toolset and select the Ride7 *ezsp_hal.rprj* project.
2. Compile the library following the instructions in [Section 2.4: Building the library on page 13](#).
3. From the **Project** menu, select **Open Project** and open the *applight\applight.rprj* Ride7 project.
4. From the **Options, Project Properties, Configuration** menu, select **ST_STM32_2_x**.
5. From the **Project** menu, select **Build Project**: a binary file sensor is produced in the *applight\app* directory.

- Note:*
- 1 *The ST_STM32_2_x configuration builds the application image using the latest available STM32F10xxx V2.0 firmware library. The ST_STM32_1_x configuration is used to build the application image using the STM32F10xxx V1.0 firmware library.*
 - 2 *For building the application image for the STR711FR2, STR912FW44 and STR750FV2 microcontrollers, select the related configurations from **Options, Project Properties, Configuration** menu.*

3.4.1 To run the light application

1. Connect the REva USB cable between the PC and the RLink.

Running steps for STM32F103x and STR7x-9x microcontrollers (Ride7 toolset)

2. From the **Options, Project Properties, Cortex RLink, Advanced Options** menu, open **Advanced Options** and select **Debug** for STM32F103x microcontrollers and unselect **Debug** for STR7x-STR9x microcontrollers.
3. From the **Debug** menu, select **Start**. The light application image is then downloaded.
4. Using the STM32F103x microcontroller, from the **Debug** menu, select **Run** and then **Terminate**.
5. Set up the serial communication channel as described in [Section 3.1 on page 14](#).
6. If using the virtual COM, reset the corresponding communication channel as described in [Section 3.1 on page 14](#).

Note: *To build and run the switch application, open the *applight\app\switch.rprj*. Follow the same steps as for the light application. Furthermore, another PC serial communication channel (serial RS232 or virtual COM through USB) is required for getting information from the switch node.*

For a description of the light and switch applications, refer to the corresponding documentation.

3.4.2 Load and run two light and switch pre-built images

Two pre-built images of the subscribed light and switch applications images for the STM32F103x microcontroller are already present in the *bin* directory. To download the *light.hex* image into the ST microcontroller Flash and run the light application, follow these steps:

1. Connect the REva USB cable between the PC and the RLink.

Steps for STM32F103x and STR7x-9x microcontrollers

2. Open the Ride7 toolset.
3. From the **Options, Project Properties, Advanced ARM Options, Processor** menu, open **Device** and select the specific STM32F103x or STR7x-9x microcontroller (STM32F103RBT6 as STM32F103x microcontroller, STR711FR2 as STR71x microcontroller, STR750FV2 as STR75x microcontroller, or STR912FW44 as STR91x microcontroller).
4. From the **Options, Project Properties, Advanced ARM Options, Debug environment** menu, open **Debug tool** and select **RLink**.
5. From the **Options, Project Properties, Cortex RLink, Advanced Options** menu, open **Advanced Options** and deselect **Debug**.
6. From the **Options, Project Properties, Cortex RLink, Advanced Options** menu, open **Advanced Options** and select **Erase target now!**
7. From the **Options, Project Properties, Cortex RLink, Advanced Options** menu, open **Advanced Options**, select **Write Target Flash Now** and choose the *bin\light.hex* image file. Wait for the image to download into the Flash.
8. Set up the serial communication channel as described in [Section 3.1 on page 14](#).
9. If using the virtual COM, reset the corresponding communication channel as described in [Section 3.1 on page 14](#).

Follow the same steps for configuring another REva board as switch node, using the *bin\switch.hex* pre-built image. Additionally, another PC serial communication channel (serial RS232 or virtual COM through USB) is required for user interaction with the switch node.

3.4.3 Building/running steps for STM32F103x and STR71x-STR91x microcontrollers using IAR toolset

1. Open the IAR toolset.
2. From the **File, Open, Workspace** menu, open the `app\light\app\IAR\light.eww` IAR workspace.
3. From the **View** menu, select **Workspace** to display supported projects. (A window appears on the left side of the IAR environment.)
4. From the **Workspace** selector window, select the application configuration according to the microcontroller to be addressed:
 - `light_STM32_2_x` configuration for the STM32F103x microcontroller with firmware library V2.0
 - `light_STM32_1_x` configuration for the STM32F103x microcontroller with firmware library V1.0
 - `light_STR71x` configuration for the STR71x microcontroller
 - `light_STR91x` configuration for the STR91x microcontroller
5. From the **Project** menu, select **Rebuild All**. An Intel HEX file is built in the `app\light\app\IAR` directory.

To download the light application

1. From the **Project** menu, select **Debug**. Wait for the image to download into the Flash memory.
2. From the **Debug** menu, select **GO**.
3. From the **Debug** menu, select **Stop Debugging**.
4. Set up the serial communication channel as described in [Section 3.1 on page 14](#).
5. If using the virtual COM, reset the relative communication channel as described in [Section 3.1 on page 14](#).

Note: When building and running the switch application, just open the `app\light\app\IAR\switch.eww` IAR workspace and follow the same steps as for the light application.

4 SPI bootloader

The EmberZNet 3.3.1 PRO stack supports the bootloader through the SPI interface feature. Basically, it is possible to update the SN260 ZigBee stack image by running a simple application on the STM32F103RBT6 microcontroller which is able to put the SN260 in bootloader mode and then sending the new stack image packets through the SPI interface. This application requires the SN260 previously loaded with a stack version (EmberZNet 3.0.1 or more recent) supporting the bootloader.

The upload process consists of the following steps:

1. Open a DOS command window.
2. Go to the directory `bin/spi_bootloader/`
3. Type the following command for loading the *STM32_upload_stack_3_3_1_spi.hex* application into the STM32F103RBT6 microcontroller Flash memory:

```
run.bat STM32 STM32_upload_stack_3.3.1_spi.hex
```

During the upload process, LEDs D7 and D6 on the REva board indicate the process status:

- LED D7 is ON: Process initialization OK
- LED D6 blinks every 2 seconds and LED D7 is ON: Upload IN PROGRESS
- LED D7 is OFF and LED D6 blinks every quarter second: Upload FAILED
- LEDs D7 and D6 blink alternatively every quarter second: Upload process completed with SUCCESS.

The user can also plug a serial cable to the REva SER1 port to receive upload status messages over the corresponding serial HyperTerminal.

The HyperTerminal settings are:

- Bit rate: 9600
- Data bits: 8
- Parity: None
- Stop bits: 1
- Flow control: None

5 Limitations and support

For any questions about the library, issues notifications refer to the specific link on the STMicroelectronics web site.

6 Revision history

Table 5. Document revision history

Date	Revision	Changes
13-Feb-2007	1	Initial release.
12-Mar-2007	2	Document updated adding full scope support for STR71xF, STR75xF and ST7LITE39 microcontrollers
02-Apr-2007	3	Document updated adding full scope support for STR71xF, STR75xF, ST7LITE39 and STR91xF microcontrollers
21-May-2007	4	Document updated adding full scope support for the EmberZNet 3.0.0 ZigBee stack
28-May-2007	5	Document updated adding full scope support for both EmberZNet 2.5.x and EmberZNet 3.x stacks
17-Sep-2007	6	Power selection jumper on the ST7LITE39 daughterboard set to 3.3V (Default configuration at delivery) and not to 5V. Added Figure 7: Boot mode setting for STR71x daughterboard on page 12 and Figure 8: Power area for REva ST7LITE39 daughter board on page 15 .
05-Oct-2007	7	Document updated adding full scope support for the most recent 3.x stack with the SPI bootloader feature. Added IAR toolset support and virtual COM through USB communication. Reviewed REva power area description and library/application building/running steps.
03-Dec-2007	8	Added information concerning Ride7 toolset and STM32F103x microcontrollers.
02-Mar-2009	9	Document updated adding full scope support for the most recent EmberZNet 3.3.1 (software version: 3.3 build 25) ZigBee PRO stack. Removed reference to the ST7LITE39 microcontroller since no longer supported.

Please Read Carefully:

Information in this document is provided solely in connection with ST products. STMicroelectronics NV and its subsidiaries ("ST") reserve the right to make changes, corrections, modifications or improvements, to this document, and the products and services described herein at any time, without notice.

All ST products are sold pursuant to ST's terms and conditions of sale.

Purchasers are solely responsible for the choice, selection and use of the ST products and services described herein, and ST assumes no liability whatsoever relating to the choice, selection or use of the ST products and services described herein.

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services it shall not be deemed a license grant by ST for the use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering the use in any manner whatsoever of such third party products or services or any intellectual property contained therein.

UNLESS OTHERWISE SET FORTH IN ST'S TERMS AND CONDITIONS OF SALE ST DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE USE AND/OR SALE OF ST PRODUCTS INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION), OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

UNLESS EXPRESSLY APPROVED IN WRITING BY AN AUTHORIZED ST REPRESENTATIVE, ST PRODUCTS ARE NOT RECOMMENDED, AUTHORIZED OR WARRANTED FOR USE IN MILITARY, AIR CRAFT, SPACE, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS, NOR IN PRODUCTS OR SYSTEMS WHERE FAILURE OR MALFUNCTION MAY RESULT IN PERSONAL INJURY, DEATH, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE. ST PRODUCTS WHICH ARE NOT SPECIFIED AS "AUTOMOTIVE GRADE" MAY ONLY BE USED IN AUTOMOTIVE APPLICATIONS AT USER'S OWN RISK.

Resale of ST products with provisions different from the statements and/or technical features set forth in this document shall immediately void any warranty granted by ST for the ST product or service described herein and shall not create or extend in any manner whatsoever, any liability of ST.

ST and the ST logo are trademarks or registered trademarks of ST in various countries.

Information in this document supersedes and replaces all information previously supplied.

The ST logo is a registered trademark of STMicroelectronics. All other names are the property of their respective owners.

© 2009 STMicroelectronics - All rights reserved

STMicroelectronics group of companies

Australia - Belgium - Brazil - Canada - China - Czech Republic - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan - Malaysia - Malta - Morocco - Singapore - Spain - Sweden - Switzerland - United Kingdom - United States of America

www.st.com

