OL2385SWUG

SIGFOX software driver

Rev. 2.1 — 6 April 2018 User guide

1 Important notice

NXP provides the enclosed product(s) under the following conditions:

This evaluation kit is intended for use of ENGINEERING DEVELOPMENT OR EVALUATION PURPOSES ONLY. It is provided as a sample IC pre-soldered to a printed circuit board to make it easier to access inputs, outputs, and supply terminals. This evaluation board may be used with any development system or other source of I/O signals by simply connecting it to the host MCU or computer board via off-the-shelf cables. This evaluation board is not a Reference Design and is not intended to represent a final design recommendation for any particular application. Final device in an application will be heavily dependent on proper printed circuit board layout and heat sinking design as well as attention to supply filtering, transient suppression, and I/O signal quality.

The goods provided may not be complete in terms of required design, marketing, and or manufacturing related protective considerations, including product safety measures typically found in the end product incorporating the goods. Due to the open construction of the product, it is the user's responsibility to take any and all appropriate precautions with regard to electrostatic discharge. In order to minimize risks associated with the customers applications, adequate design and operating safeguards must be provided by the customer to minimize inherent or procedural hazards. For any safety concerns, contact NXP sales and technical support services.

Should this evaluation kit not meet the specifications indicated in the kit, it may be returned within 30 days from the date of delivery and will be replaced by a new kit.

NXP reserves the right to make changes without further notice to any products herein. NXP makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. Typical parameters can and do vary in different applications and actual performance may vary over time. All operating parameters, including Typical, must be validated for each customer application by customer's technical experts.

NXP does not convey any license under its patent rights nor the rights of others. NXP products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the NXP product could create a situation where personal injury or death may occur.

Should the Buyer purchase or use NXP products for any such unintended or unauthorized application, the Buyer shall indemnify and hold NXP and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or

unauthorized use, even if such claim alleges NXP was negligent regarding the design or manufacture of the part.

2 Introduction

The main intention of this user guide is to enable engineers to set up OL2385 based SIGFOX boards for testing. This user guide explains how to use the OL2385 transceiver for rapid prototyping of microcontroller based SIGFOX applications. It also contains a basic introduction to the SIGFOX ecosystem, an explanation regarding necessary registration of devices at SIGFOX network for demo evaluation and basics of SIGFOX P1 certification testing. This documentation also describes how to install and use the SIGFOX hardware/software setup.

The OM2385/SF001 development kit is an evaluation platform based on the OL2385 chip. See the OL2385 data sheet for detailed information.

3 SIGFOX network architecture

SIGFOX is an operated telecommunication Low-Power Wide-Area (LPWA) public network, dedicated to the Internet of Things. This telecommunication technology has been created with the sole focus on small messages that are transmitted on the radio only very few times per day (12 bytes messages uplink up to 140 times per day and 8 bytes messages downlink up to 5 times per day). It is not appropriate for high-bandwidth applications, such as multimedia and permanent broadcast. The SIGFOX network operates at sub-GHz frequencies on ISM bands, for example, 868 MHz in Europe/ETSI and 902 MHz in the US/FCC. This Ultra-Narrow Band (UNB) modulation based technology has 162 dB budget link, which enables long range communications. These SIGFOX messages, containing a payload of up to 12 bytes, are transmitted by a radio transceiver chip into the SIGFOX network, which is then received by a SIGFOX base station. NXP plays the role of this transceiver chip provider in the SIGFOX ecosystem.

There are four roles in the system:

- **Modem manufacturers** provide transceiver chips running with a SIGFOX software protocol stack and identified by a unique id/private key (used for encrypting the radio messages). These transceivers will be integrated in final modules that are sold to end customers. The modules must have a network registration containing the following information:
  - ID
  - Porting authorization code (PAC), one time use only
  - Certificate number given by the product manufacturer (P_xxxx_xxxx_xx)

The trio of key, ID and PAC is provided by SIGFOX to modem manufacturers. These manufacturers flash this information in nonvolatile memory of their IC during production. The ID and PAC can be retrieved by the user and thus are sharable entities. However, the encrypted key should not be shared, visible, or retrievable by anyone. It must reside inside a protected area of memory where it is only readable by the firmware for encryption purposes only. It should not be writable by anyone after production.

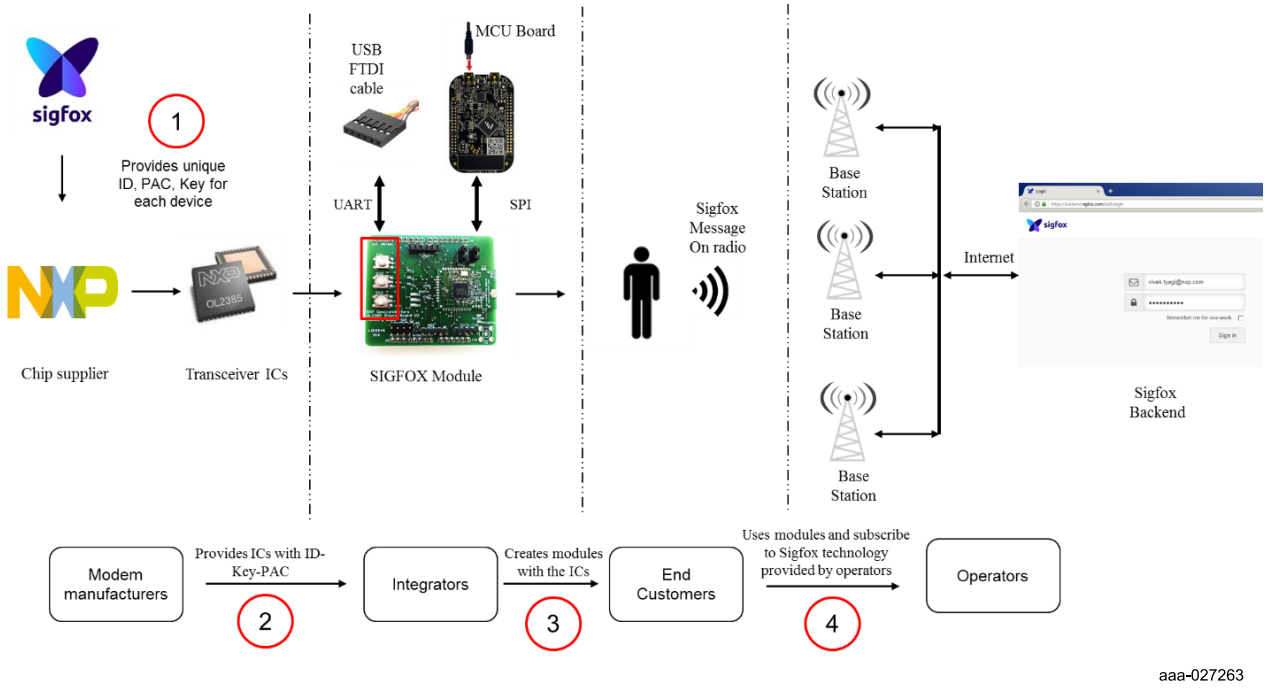


Figure 1. SIGFOX Network Architecture

- **Operators** operate their SIGFOX networks (base stations and backend system).
- **Integrators** build end-user systems or modules that will have manufacturer's transceivers on them with a SIGFOX protocol stack inside.
- **End customers** buy and use these systems or modules that include SIGFOX transceivers. each end customer will also have to buy a subscription to SIGFOX technology from an operator.

4 External/User interfaces to OL2385

4.1 Host MCU interface through SPI

A 5-wire SPI connection is defined between OL2385 board and Host microcontroller where host is configured to be the Master and always provides the clock. OL2385 is initiated in Pseudo-Slave mode. The five essential lines required for such interface is shown in Figure 2.

Reason for extra pin Ack

The host microcontroller can be sometimes faster than the slave device and can send the data while slave is not yet ready to receive. This might result in communication error due to nonsynchronization. For example, at slave device few initial clocks could be missed from the host, because it was not ready to receive. Therefore, each of the frame transfers between host and OL2385 is completely controlled via a software driven chip select CS and Ack pins. The protocol and timing diagram makes sure that the handshake is properly done and none of the bytes are missed, as shown in the following sections.

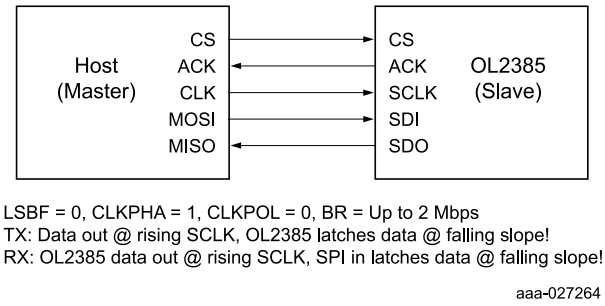


Figure 2. 5 wire SPI diagram

A messaging protocol has been defined for SPI interface between OL2385 and host microcontroller. The protocol governs the states of each of the five SPI interface pins, as shown in Figure 3. **It is very important to note here that Host is the master and always the initiator of the communication, which means that there cannot be an independent transfer of bytes from OL2385 to host.** The communication will take place always by transfer of bytes from host to OL2385, and if a response is required back only then data transfer will take place from OL2385 to host. The cases in which such a response is desired is explained in further in Section 6.2.4.4 "SPI command descriptions". The sequence of pin driving should take place strictly in the following manner:

Note: By default, after a reset, OL2385 goes to low-power mode with CS pin configured as wakeup pin. Before proceeding to the low-power mode, OL2385 configures all its pins to pulled-down configuration, except for CS and Ack pins of SPI. CS and Ack pins are pulled high. For the low-power mode to work properly without any current leakage, it is important to have the same settings of the pins on MCU side. All pins on MCU should be pulled low and only CS and Ack should be high.

1. Initially, the CS and Ack pins should be driven high on the MCU side. OL2385 as a slave waits for the CS pin to be driven low by the host to start communication.
2. Host drives CS pin low to signal data transfer.
3. OL2385 drives Ack pin low to signal that it is ready for clocking and data transfer.
4. Host provides clock and writes data on MOSI
5. **OL2385 reads the first byte of data. This is the length byte that describes how many valid data bytes are to be followed on SDI, including itself.** After receiving all the bytes, OL2385 drives the Ack pin high to signal data is received successfully.
6. Host drives CS back to high again to signal data transfer from host to OL2385 is complete.

The data is now decoded at OL2385 and processed according to its meaning. If a response is required to be submitted back to host after processing, the following sequence should then be initiated from step number 7:

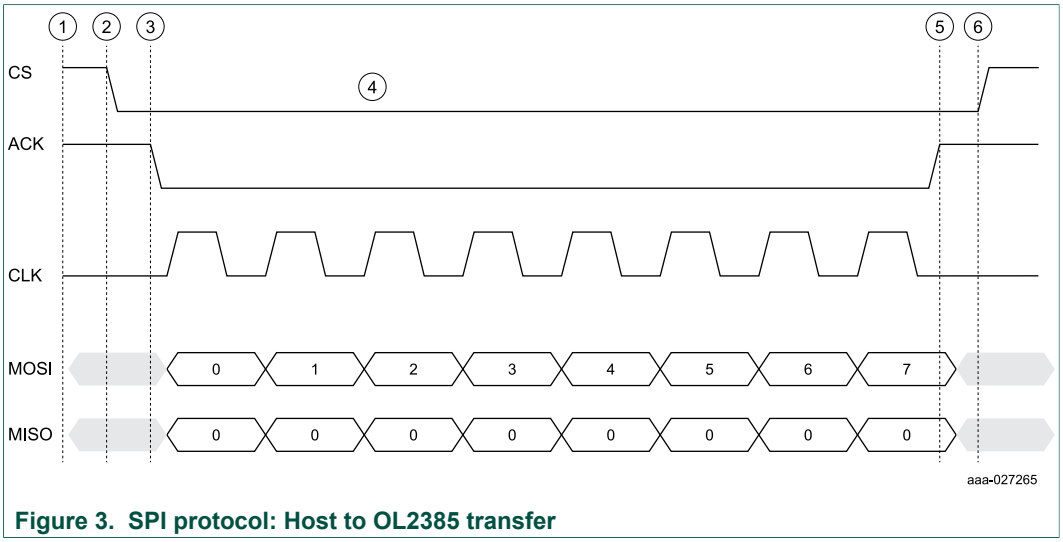


Figure 3. SPI protocol: Host to OL2385 transfer

1. CS and Ack pins are high after data transfer from host to OL2385.
2. OL2385 drives Ack pin low to signal the host that OL2385 has data to transfer as response.
3. Host drives CS pin low to signal host is ready to provide clock for such a transfer.
4. Host provides clock and data is written on SDO by OL2385.
5. **Host reads the first byte of data. This is the length byte that describes how many valid data bytes are to be followed on SDO, including itself.** After receiving all the bytes on MISO, the host waits for the Ack pin to be driven high and OL2385 drives the Ack pin high to signal that data is now completely sent.
6. Host drives CS back to high again to signal data reception is complete from OL2385 to host.

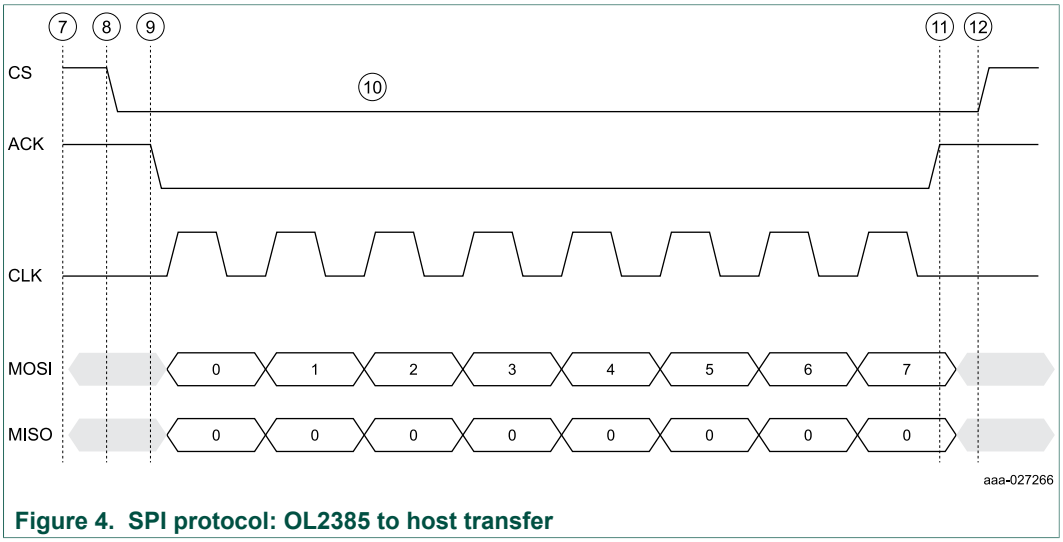


Figure 4. SPI protocol: OL2385 to host transfer

SPI commands

The bytes received or sent over the SPI interface must also follow a predefined format and order as per the SPI protocol. The bytes sent from host to OL2385 are called as command frames or information frames. The bytes sent from OL2385 to host are called as Ack frames. The format follows a similar pattern for both of these types of frames, as shown in Figure 5. The command frames contain a command from a set of allowed

commands coded in the second byte, which is called *Command code*. An action is taken by OL2385 as per the designed meaning of such code.

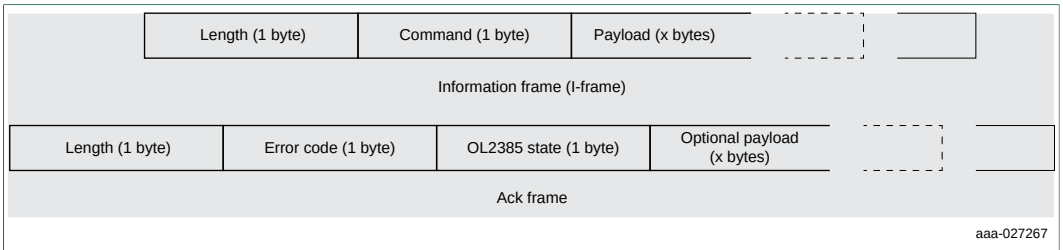


Figure 5. SPI frame types

The first byte of each frame type indicates the length of the frame to be received, including itself. A length of 4 means 3 more bytes are going to follow after the first byte. The payload field in both types of frames is optional. For details of each command, refer to the SIGFOX SPI Command Interface Description.xls file at:

www.nxp.com/products/software-and-tools/hardware-development-tools/freedom-development-boards/om2385-sf001-ol2385-wireless-sub-ghz-transceiver-sigfox-development-kit-with-kl43z:OM2385-SF001?tab=Documentation_Tab

The data bytes between OL2385 and the host do not only just follow the format, they also follow a particular sequence, depending on the state of the device, as shown in the message sequence chart in Figure 6.

Data transfer protocol over SPI

Because the frequency of Sigfox messages is very low per day, the OL2385 device generally remains in power-down SLEEP mode under normal conditions.

1. The device is always awakened by sending the Send wakeup (2 byte, fixed length) frame from the host master.
2. On receiving such frame, an OL2385 device performs a wake-up reset and moves to a ready state in 100 milliseconds.
3. The host should wait for the 100 ms wake-up to complete before sending the further frames. No acknowledgment is sent from OL2385 after wake up.
4. Once the device wakes up, it is ready to receive command frames from the host.
5. An Information frame or I-frame is used to send a command to OL2385.
6. An Ack frame is used to send back a response, if required.

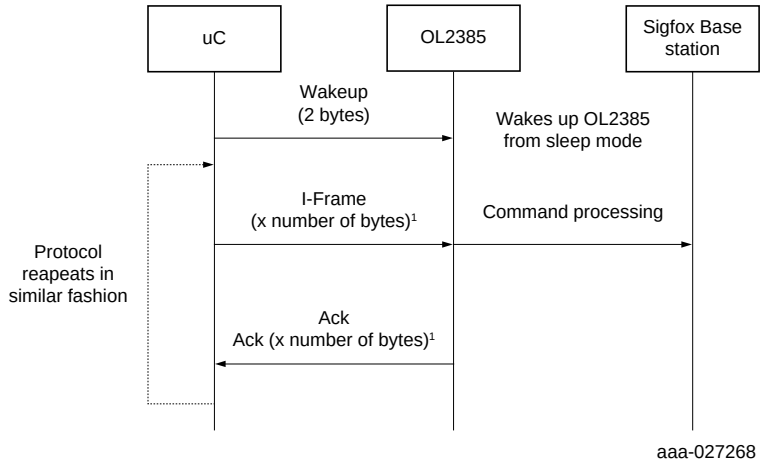


Figure 6. Message Sequence Chart SPI protocol

4.2 Button pressed based interface

To be implemented fully in software

4.3 UART interface

To be implemented fully in software

5 Setting up the hardware

The connection of any OL2385 board with any microcontroller is based on connecting the SPI pins, power supply pins and RESET pin. The positions of SPI pins on OL2385 is fixed and explained in the following sections. The positions of SPI and power pins on any other microcontroller board has to be read out from its corresponding user guide. An example of such connection with KL43Z pins is given below. The NXP reference design is based on the Arduino style of pin layout. Therefore, NXP reference design boards can be plugged directly on top of a KL43Z board as shown in Figure 11. The SPI, RESET and Power pins will match automatically. If your microcontroller is not Arduino style, you will have to connect the pins of the OL2385 with wires to your microcontroller's corresponding pins.

5.1 Overview of the OM2385/FS001 development kit

The OM2385/FS001 development kit provides an evaluation platform for designing SIGFOX network applications that use NXP's OL2385 single-chip RF transceiver. The kit consists of three boards: an OL2385 shield board, OL2385 reference design board and FRDM-KL43Z board. The OL2385 reference design board is permanently affixed to the surface of the OL2385 Shield Board. The reference design board contains an embedded OL2385 transceiver and serves as a wireless modem.

When connected to an antenna, included in the kit, the board provides all the functionality required to communicate with the SIGFOX network. The OL2385 shield board contains connectors for external communication. The Shield Board is mounted by means of four Arduino connectors to the FRDM-KL43Z. The FRDM-KL43Z acts as the communication

link between the development kit and a PC. It comes preloaded with microcode that manages the interface between the PC and the OL2385 reference board. Users must initially register their device with SIGFOX using a unique ID and access code provided with the kit. Once the device has been registered, the kit can be used to connect to the SIGFOX network and test the functionality of the OL2385-based application under development.

To interact with the development kit, users must connect the kit to a PC through the OpenSDA port on the FRDM-KL43Z. A terminal emulator, such as HyperTerminal, provides the interface, allowing users to log in to the network and send and receive messages. Designers can also use the Kinetis Design Studio (KDS) or MCUXpresso IDE to develop and download microcode to the KL43Z.

5.2 Getting started

NXP's analog product development boards provide an easy-to-use platform for evaluating NXP products. The boards support a range of analog, mixed-signal and power solutions. They incorporate monolithic ICs and system-in-package devices that use proven high-volume technology. NXP products offer longer battery life, a smaller form factor, reduced component counts, lower cost and improved performance in powering state-of-the-art systems.

The tool summary page for the OM2385/SF001 development kit is located at www.nxp.com/OM2385. The **Overview** tab provides an overview of the device, product features, a description of the kit contents, a list of (and links to) supported devices, list of (and links to) any related products and a Get Started section.

The **Get Started** section provides links to everything needed to start using the device and contains the most relevant, current information applicable to the FRDM-PF1550EVM.

- Go to www.nxp.com/OM2385
- On the **Overview** tab, locate the **Jump To** navigation feature on the left side of the window.
- Select the **Get Started** link.
- Review each entry in the **Get Started** section and download an entry by clicking on the title.
- After reviewing the **Overview** tab, visit the other product related tabs for additional information:
  - **Documentation**: download current documentation
  - **Software & Tools**: download current hardware and software tools
  - **Buy/Parametrics**: purchase the product and view the product parametrics

After downloading files, review each file, including the user guide which includes setup instructions. If applicable, the bill of materials (BOM) and supporting schematics are also available for download in the **Get Started** section of the **Overview** tab.

5.2.1 Kit contents/packing list

The OM2385/SF001 development kit contents include:

- Assembled and tested OM2385/SF001 FRDM board mounted to a firmware-loaded FRDM-KL43Z board
- Antenna with attached micro-FL connector
- Standard A (male) to mini B (male) USB cable

- Quick start guide

5.2.2 System requirements

The kit requires the following to function properly with the software:

- USB enabled computer running Windows XP, Vista, 7, 8, or 10 (32-bit or 64-bit)
- Terminal emulation software (such as HyperTerminal)

5.3 Getting to know the hardware

5.3.1 SF001 board overview

The OM2385/SF001 consists of a base board (the OL2385 shield board) with a permanently attached daughter board (the OL2385 reference design board). The combination, along with the attached FRDM-KL43Z board, serves as a development platform that provides wireless modem access to the SIGFOX network. Once properly registered, the board allows users to send and receive messages across the network.

5.3.2 Board features

The board features are as follows:

- Arduino connector compatibility with other Freedom boards
- Support for UART, SPI, MDI and GPIO communication
- SIGFOX Communication Library

5.3.3 OM2385/SF001 Block diagram

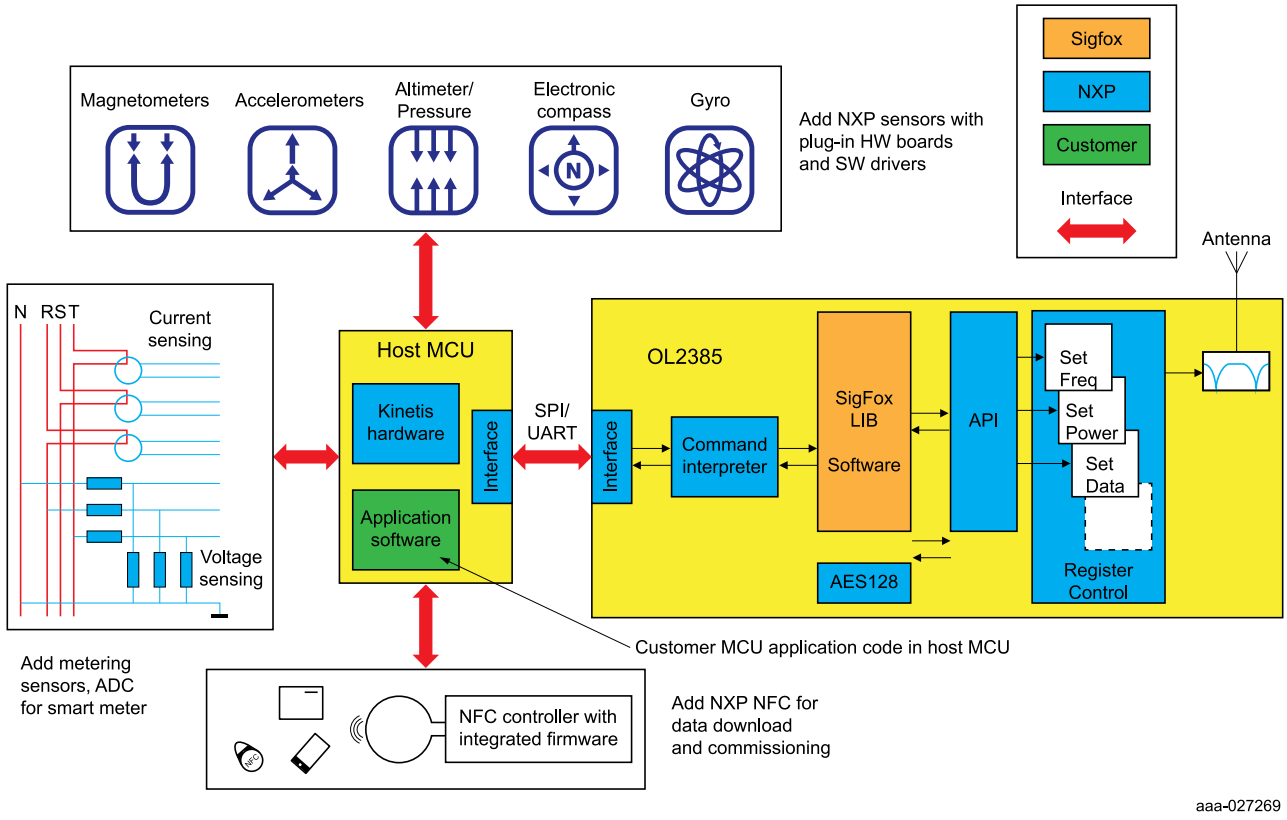


Figure 7. Block diagram

5.3.4 OL2385 reference design: Board description

Figure 8 describes the main elements on the OM2385/SF001 board.

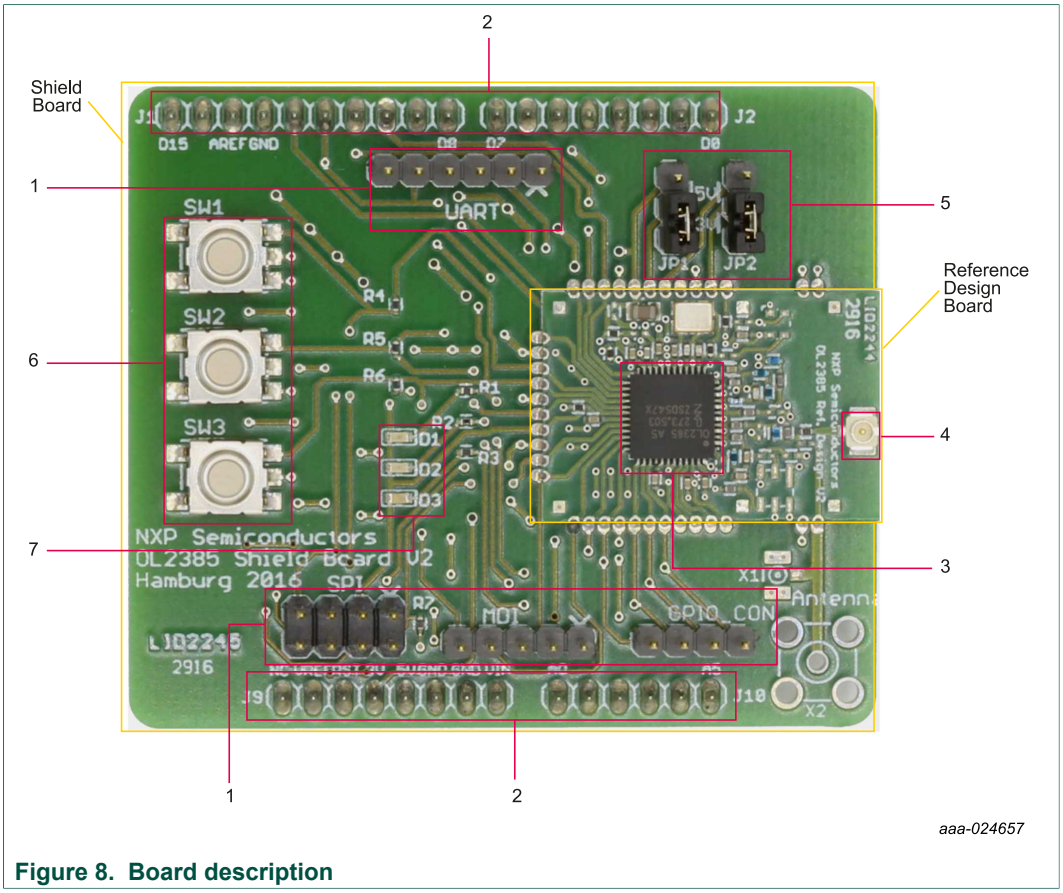


Figure 8. Board description

Table 1. Board description

Number	Name	Description
1	Communication connectors	Provide connectivity for SPI, MDI, GPIO and UART support
2	Arduino connectors	Provide connectivity to FRDM-KL43Z and other Freedom boards
3	OL2385	Low-power multichannel UHF RF wireless platform
4	SMA connector	Provides connectivity to UHF antenna
5	Jumpers	Select board voltage levels
6	Button switches	Control digital inputs to Arduino connectors
7	LEDs	Indicate status

5.3.5 OL2385 ref design board jumper definitions

Figure 9 shows the location of jumpers and switches on the OM2385/SF001 evaluation board.

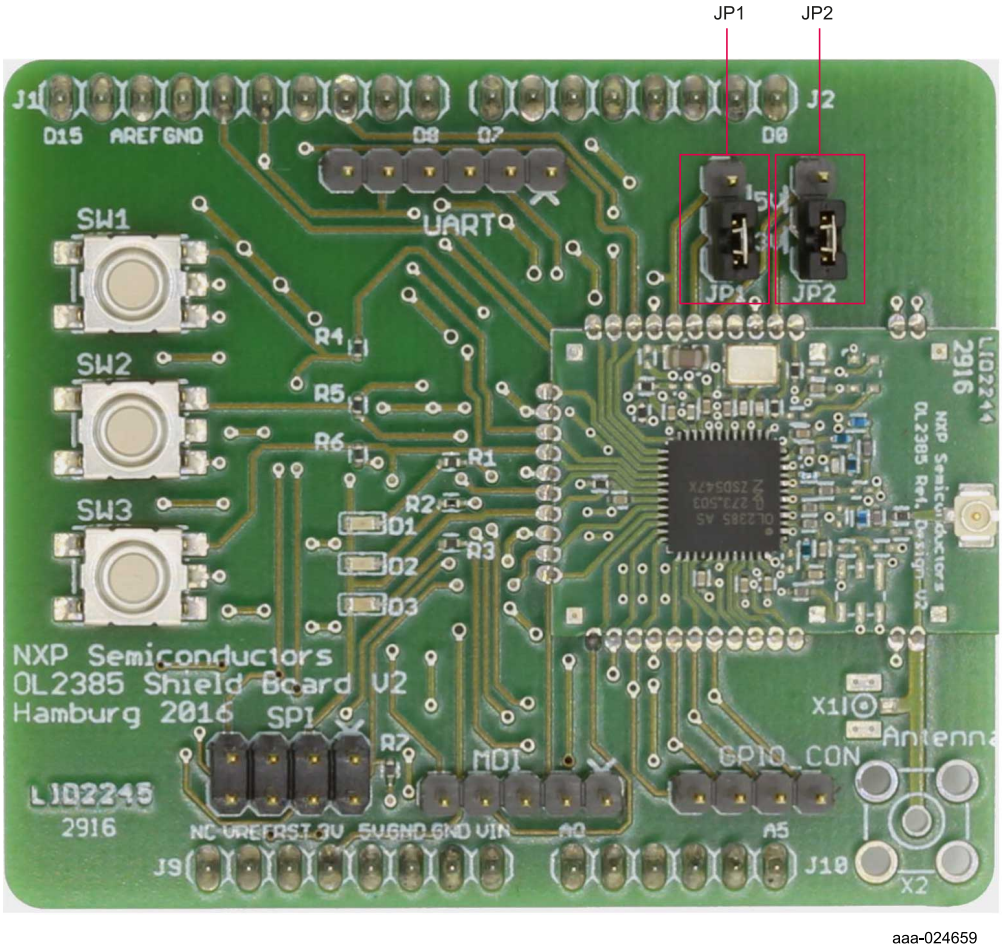


Figure 9. Jumpers

5.3.6 OL2385 ref design board switch definitions

Figure 10 shows the location of switches on the OM2385/SF001 Shield Board.

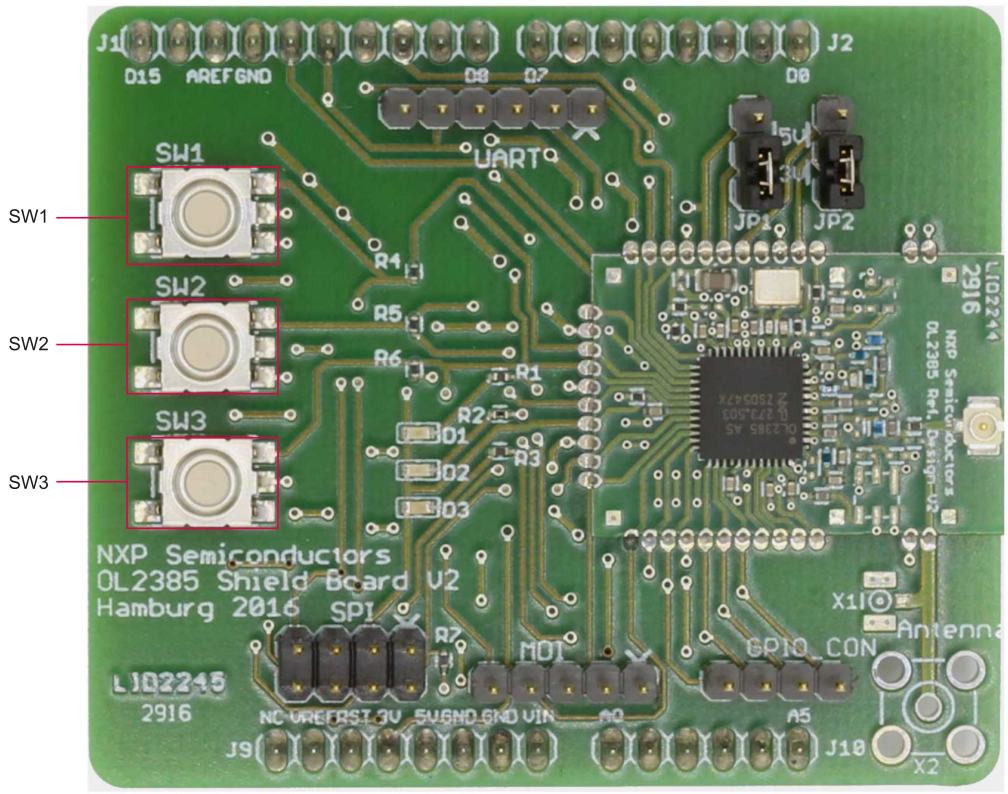


Figure 10. Switches

5.4 Connecting OL2385 with MCU

The Arduino based NXP reference design boards (green board) connect to a KL43Z board as shown in Figure 11. Buttons on the OL2385 board are facing toward the USB ports on the KL43Z board. Connect your NXP reference board in a similar manner to the KL43Z. If you have a different type of OL2385 based board, refer to the later sections in this document, specifically the SPI pins of OL2385. Connect the boards accordingly.

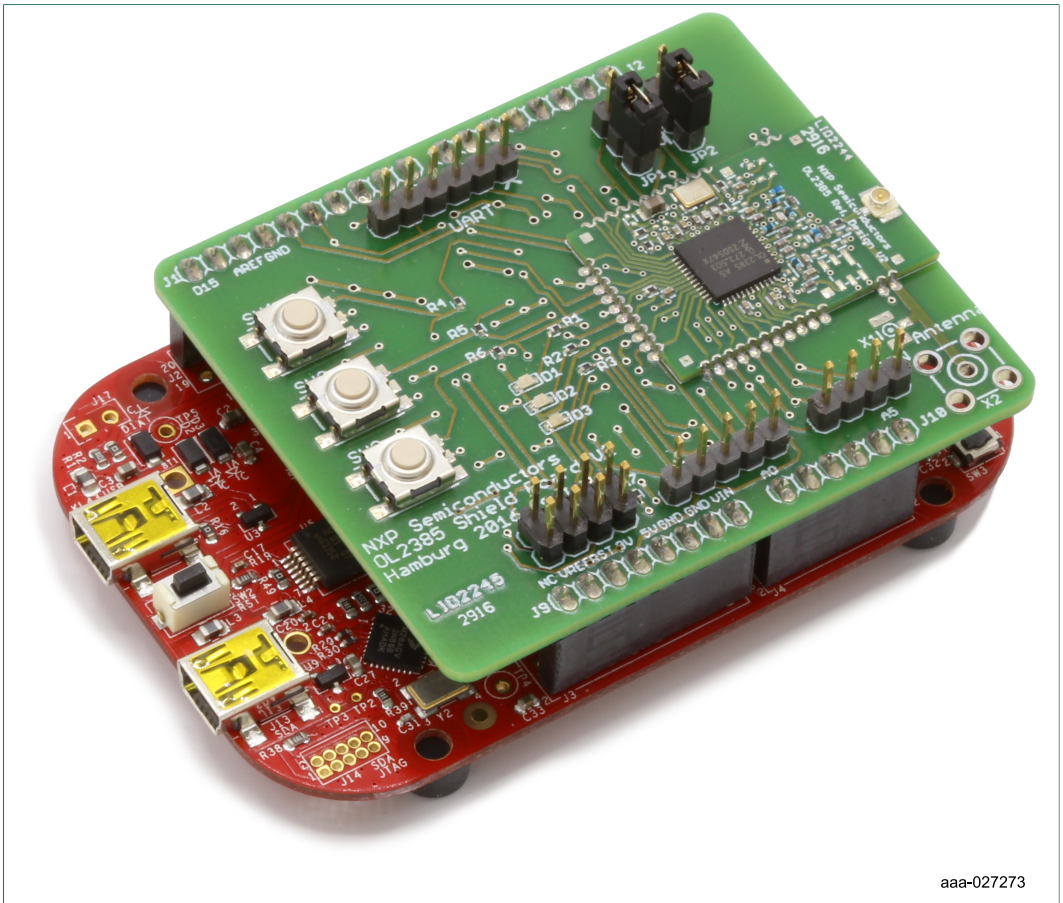
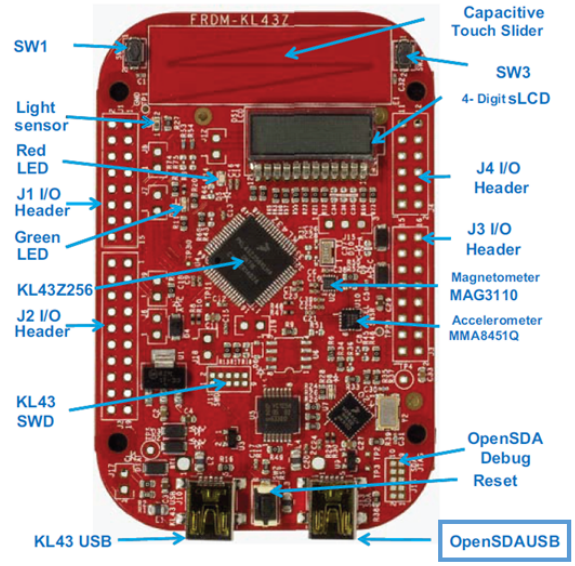
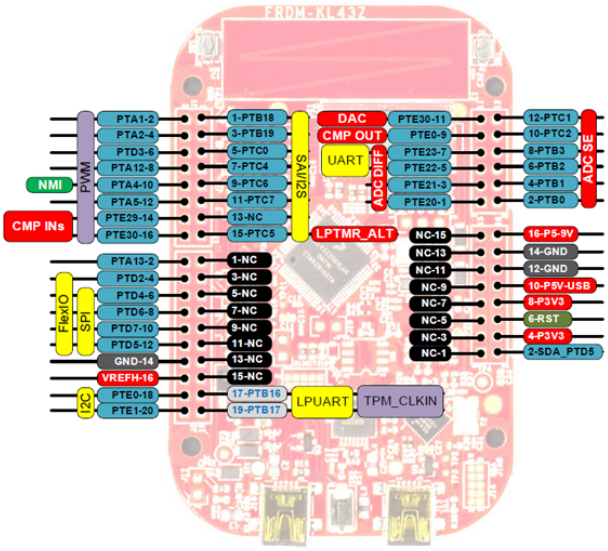


Figure 11. HW connection using KL43Z and NXP reference board

The KL43Z board running with NXP SIGFOX testing firmware spi_sigfox_demo.srec executable file has SPI pins mapped on J2 I/O header, which is marked in Figure 12. The exact pin numbers are specified in Table 2. The power supply and RESET pins are mapped on the J9 I/O Header.





Supply source	Valid range	OpenSDA operational?	Regulated on-board?
OpenSDA USB	5 V	Yes	Yes
KL43 USB	5 V	No	Yes
V_{IN} Pin	4.3 – 9 V	No	Yes
3.3 V Pin	1.71 – 3.6 V	No	No
Coin cell battery	1.71 – 3.6 V	No	No

Pin function	KL43Z48M pin
Sigfox SPI MOSI	PTD6
Sigfox SPI MISO	PTD7
Sigfox SPI SCK	PTD5
SIGFOX SPI CS	PTD4
SIGFOX ACK	PTD2

aaa-027274

Figure 12. KL43Z pin diagram

Table 2. Example pin connections with KL43Z board

OL2385(fixed)	KL43Z	Purpose
P15	PTD 0 (Configured as output at KL43Z)	CS (pin polled by OL2385 for incoming data)
P17	PTD 5 (Configured as input at KL43Z)	ACK
P16 (SDI)	PTD2 MOSI	Data line
P14 (SCK)	PTD1 SCK	Clock
P13 (SDO)	PTD3 MISO	Data line
RSTN	RESET/PTA20	Reset pins
Power Supply	P3V3	Power supply pin
GND	GND	Ground pin

5.4.1 Setting up KL43Z board with terminal program

A new KL43Z board needs the following steps to be performed to get it up and running. To configure the OM2385/FS001, the user must:

1. Download drivers for the FRDM-KL43Z (first time only).
2. Set up and configure terminal program to control OL2385 using KL43Z (first time only).
3. Connect the hardware for use with the SIGFOX network.

5.4.1.1 Downloading and installing the driver for the FRDM-KL43Z

This procedure involves downloading the FRDM-KL43Z driver from the P&E Microcomputer Systems website and installing it on the host PC.

1. Go to the P&E Microcomputer Systems OpenSDA page at www.pemicro.com/opensda and, in the Windows USB Drivers box, click to download the PEDrivers_install.exe file to a location on the host PC.
2. When the download completes, click on the PEDrivers_install.exe file and follow the instructions to install the driver.
3. Connect a USB cable between the host PC and the FRDM-KL43Z USB port labeled SDA (J13).
4. Open Windows Explorer on the host PC. An icon labeled FRDM-KL43Z appears as a removable drive on the PC.

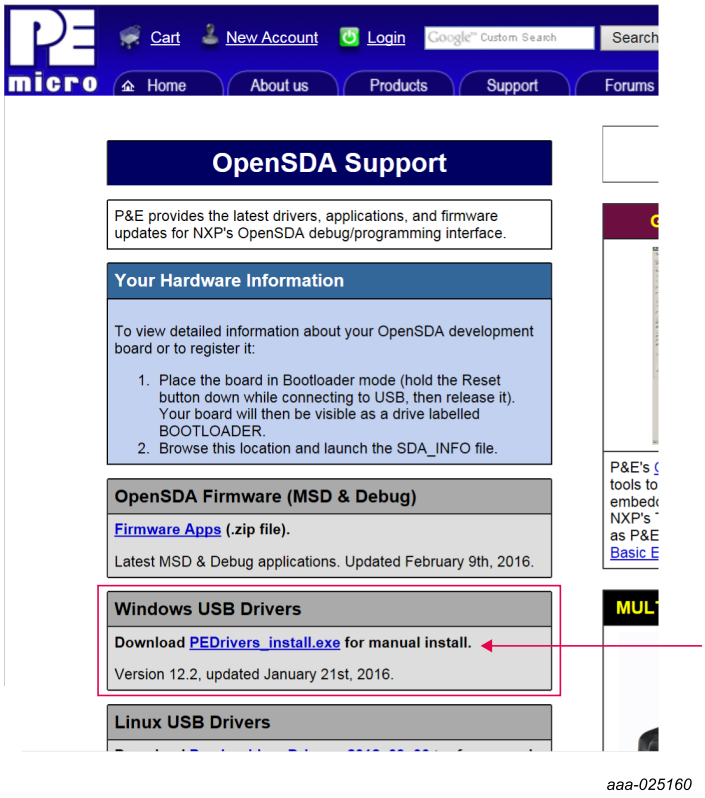


Figure 13. Download the PEDrivers_install.exe file to a location on the host PC

The FRDM-KL43Z is now ready for use with the OM2385/SF001 development kit.

5.4.1.2 Connecting the hardware for use with the SIGFOX network

To connect the hardware to send messages across the SIGFOX network, do the following:

1. Check to assure that the OM2385 board is firmly attached to the FRDM-KL43Z. When connecting the boards, the three switches on the shield board should be on the same side as the USB ports on the FRDM-KL43Z board.
2. Attach the PCB antenna (included with the kit) by snapping the uFL connector on the antenna to the uFL connector on the shield board.

3. Connect the Standard A end of the supplied USB cable to a Windows host PC. Connect the Mini B to the FRDM-KL43Z USB port labeled SDA (J13).

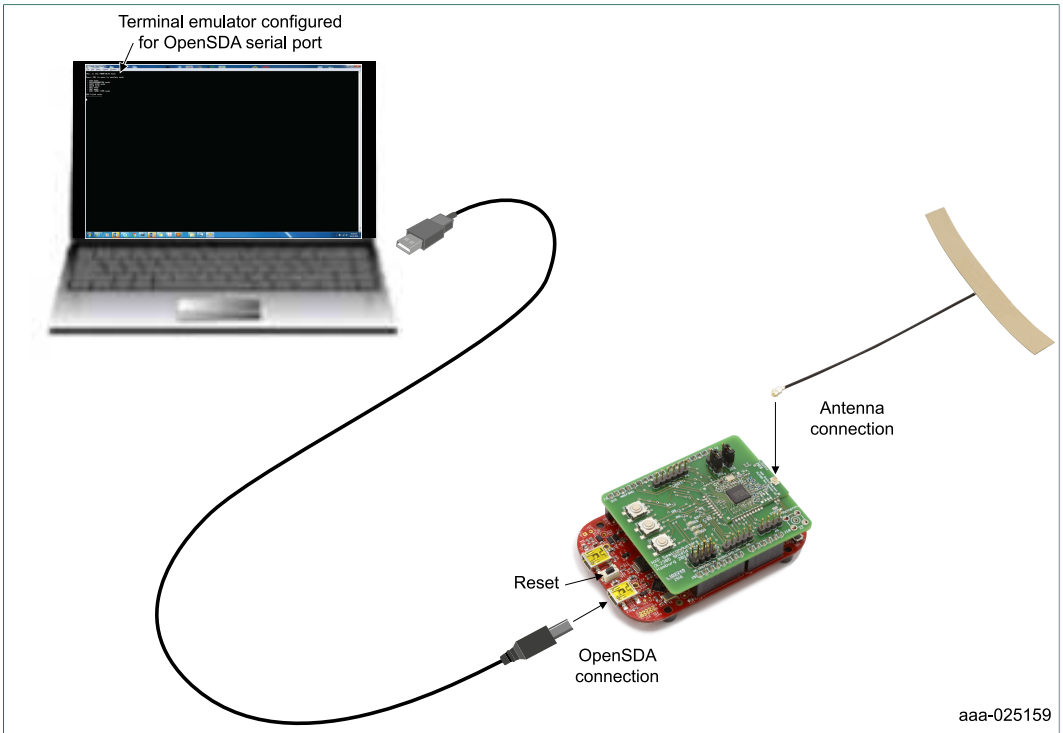


Figure 14. Connecting the hardware

The OM2385/SF001 is now ready to be configured for use with the SIGFOX network.

5.4.1.3 Setting up terminal program

1. Plug one end of a mini USB cable to the SDA USB port on the board. Plug the other end of the cable to a PC. See Figure 14.
2. For the first time, install drivers for this device on a PC. Install PEDrivers_install.exe downloaded from the page www.pemicro.com/opensda/.
3. If the device has an updated bootloader installed in it already, it will be shown detected as a removable device on windows PC labeled as FRDM-KL43Z as shown in Figure 15. If the bootloader is not installed, follow steps given in Quick Start Guide for FRDM-KL43Z (www.nxp.com/FRDM-KL43Z) to update bootloader.
4. If the KL43Z is not preflashed with a srec file, download the file from the web else move to next step. Now drag the file into this removable drive as shown above. Do not open the removable drive and copy-paste, but just drag the file on top of it. Press copy and replace, if you are updating the software.
5. Press the reset button located between the two USB ports to let the device accept and run the updated software inside the KL43Z.
6. Install and open a terminal software like Tera Term with the settings as shown in Figure 15. Use the OpenSDA com port number, which you can get from the Windows Device Manager program. Press and hold (or right-click) the **Start** button, then select **Device Manager** from the context menu. In the Device Manager, click on **Ports (COM & LPT)**. Under OpenSDA – CDC Serial Port, note the COM port number. You can set these settings on the **Setup** menu of the terminal program.

Note: Understand that the COM connection is established as soon as you select the COM port and click on **OK**. If you remove the USB cable without closing the terminal, the COM connection breaks but is not fully closed. And if you plug the cable back in the USB, the COM connection does not establish again. Therefore, you will see that the terminal does not respond. **Therefore, always close the terminal before removing the USB cable and open the terminal only after you have plugged back in the USB cable.** Connect the RESET pins of both boards and use middle reset button on the KL43Z board to reset. Unplugging and replugging the USB is not the preferred way to reset the boards.

7. After you Press the reset button on the KL43Z, a menu as shown in Figure 15 will appear, indicating that the KL43Z board is properly set up.
8. Type **1** and press Enter to send wakeup command over SPI. By default, after power-on reset, the device goes in deep sleep or POWEROFF2 power-saving mode. **Therefore, the wakeup command always has to be sent first after any kind of board reset.**
9. Type **8** and press Enter to send a Get info command. This command will display the ID and the PAC value of the device. Even if these values are zero, if you see SPI TX buffer and SPI RX buffer bytes, SPI bidirectional communication is working.
10. The next command depends on the region you would like to test, namely RCZ1, RCZ2, RCZ3 or RCZ4. Choose between 0x15, 0x16, 0x17 or 0x18 accordingly. Note that the OL2385 boards for each of these respective regions are different. For example: RCZ2 and RCZ4 uses external PA whereas, RCZ3 uses 27.6 MHz external TCXO. However, the OL2385 firmware supports all four regions. However, the system as a whole will only work if you choose the appropriate region on the terminal for the hardware.
11. The rest of the commands are self-explanatory and depend on your choice of testing tool. For example, a Spectrum analyzer might need a Continuous wave command, SIGFOX base station might need a Send payload command or SIGFOX SFXIQ P1 certification tool might need a Send test mode command. If required, refer to OL2385_SPI_Interface_Commands.xlsx for details about SPI commands located at www.nxp.com/OM2385

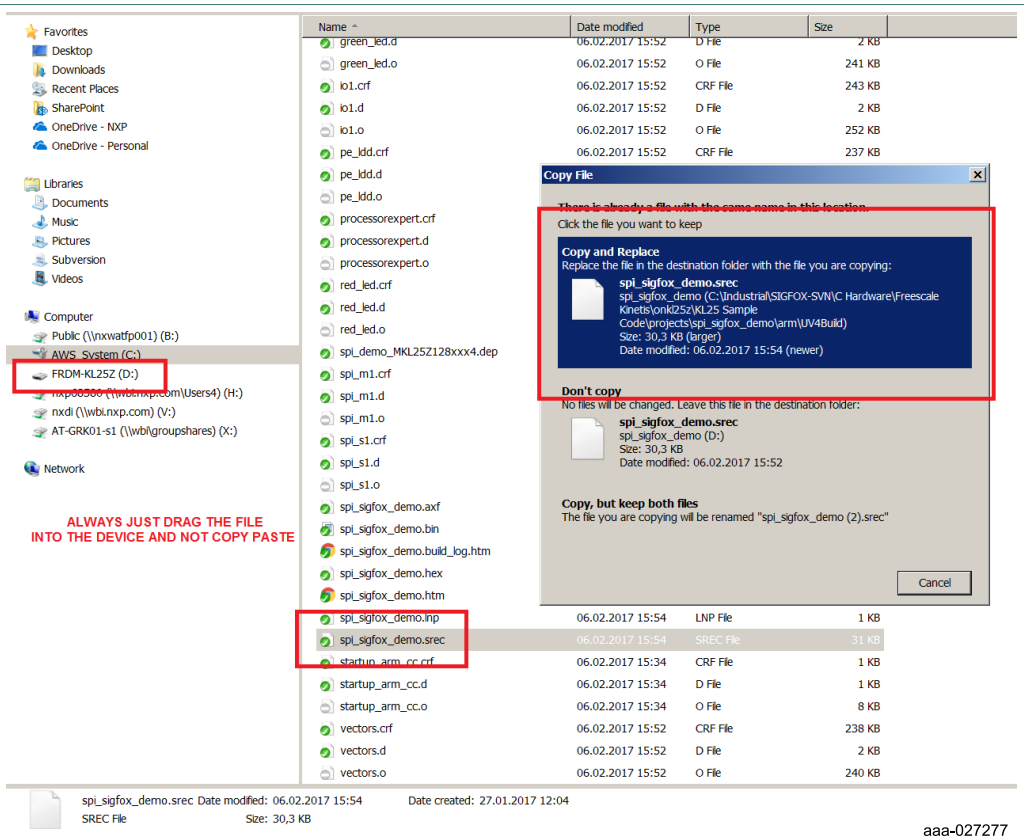


Figure 15. Flashing the firmware on KL43Z

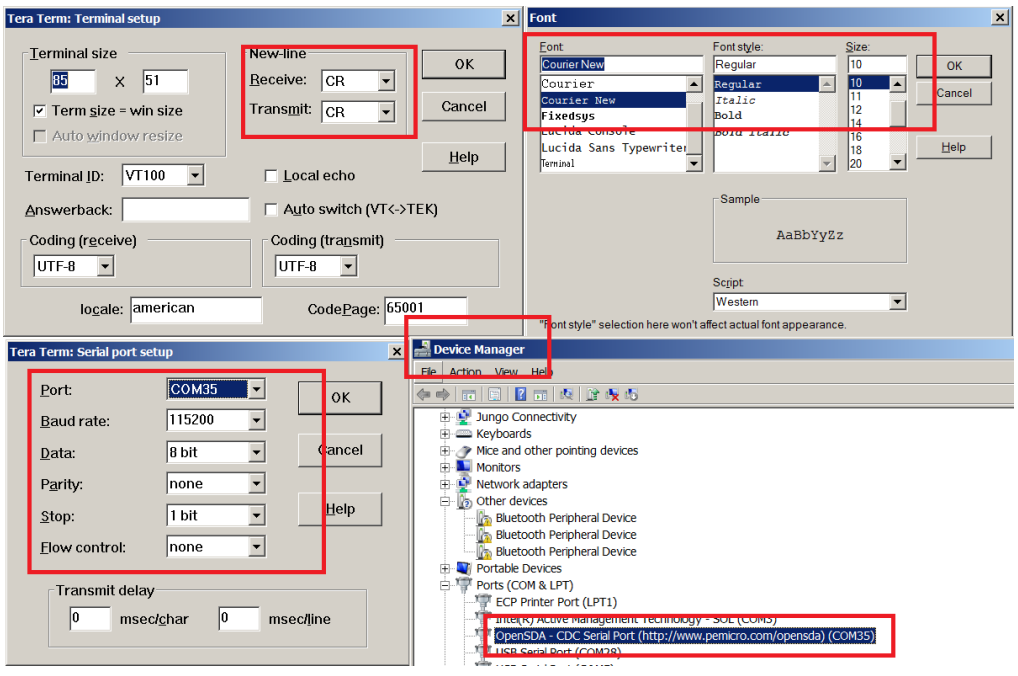


Figure 16. Tera-term example settings

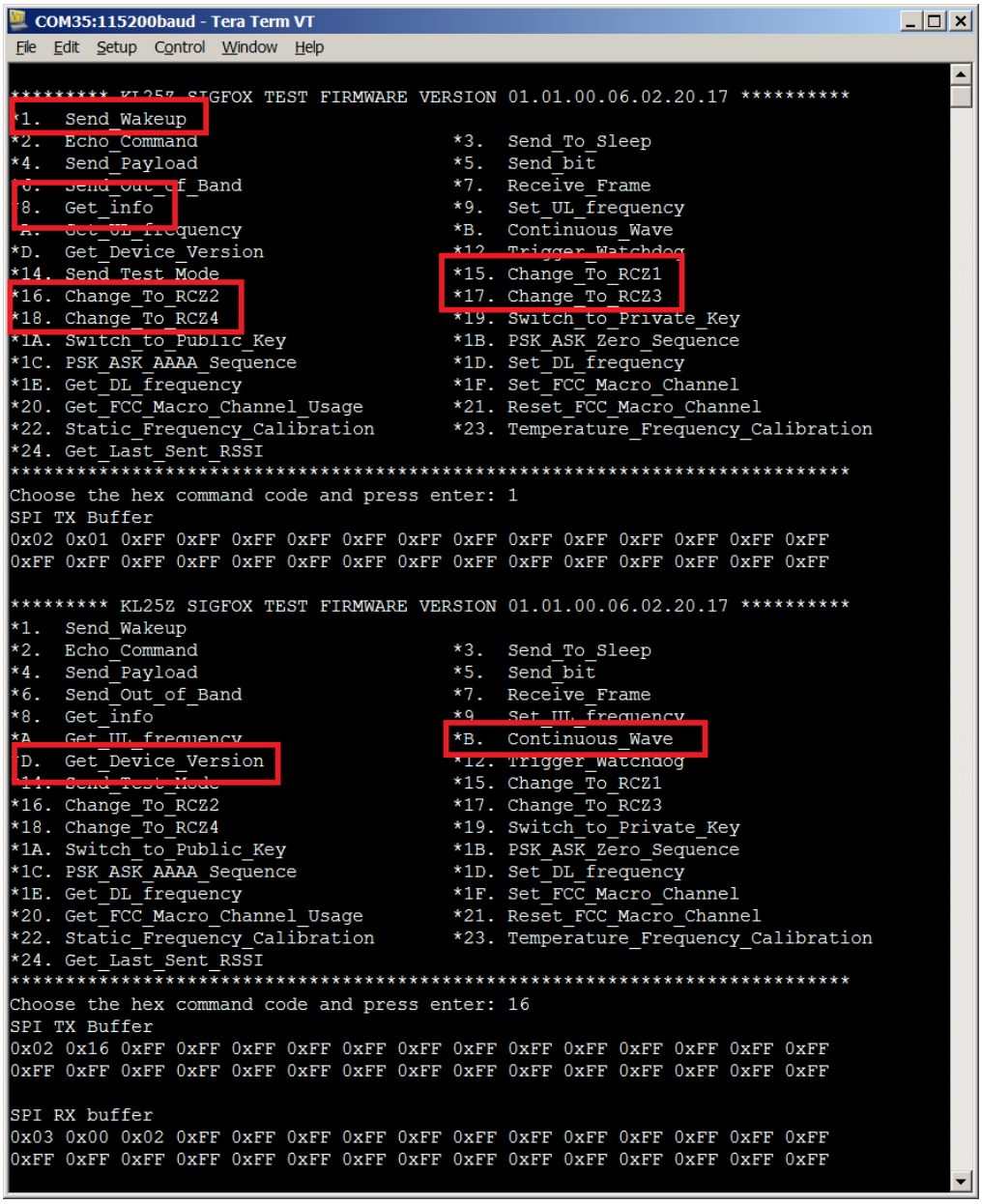


Figure 17. KL43Z MCU using terminal program

5.4.2 Flashing the OL2385 binary

Flashing the OL2385 binary

If the OL2385-based board is not preflashed with firmware or the firmware needs to be updated, the following are required before flashing the board.

1. OL2385 based module board with an MDI interface. Check the schematics or search for an MDI label on the hardware.
2. Hex file to be flashed, if the board is not yet flashed.
3. MRK III 2-link box to flash hex file.

4. MRK III 2-link box driver installation package.
5. Batch file for flashing the hex file.
6. Batch file for powering up the board.

Before flashing the hex file, install the drivers for the 2-link box on your PC. The installation instructions are provided in the 2-Link Installation files folder provided within the Customer_Support_Package.zip file. Follow these steps for flashing:

1. Connect the 2-link box as shown in Figure 18. Find the MDI interface on the board and connect the ground wire (brown) of the 2-link box to the marked with a white arrow.
2. Place the hex file, MtGV0-SIGFOX.hex, in the **Customer_Support_Package/OL2385** folder. If the hex file name is different, rename the file to MtGV0-SIGFOX.hex. This is the name of the file used inside the batch file script.
3. The MRKIIIlink.exe and MRKIIIlink.dll should be present in **Customer_Support_Package** folder provided by NXP.
4. Run the OL2385_load_hexfiles.bat batch file to write the hex file software on OL2385.
5. Run the OL2385_powerup_device batch file to power up the device by the 2-link box, if you do not have any other power supply connected like microcontroller board or external battery. Otherwise, you can now disconnect the 2link box.

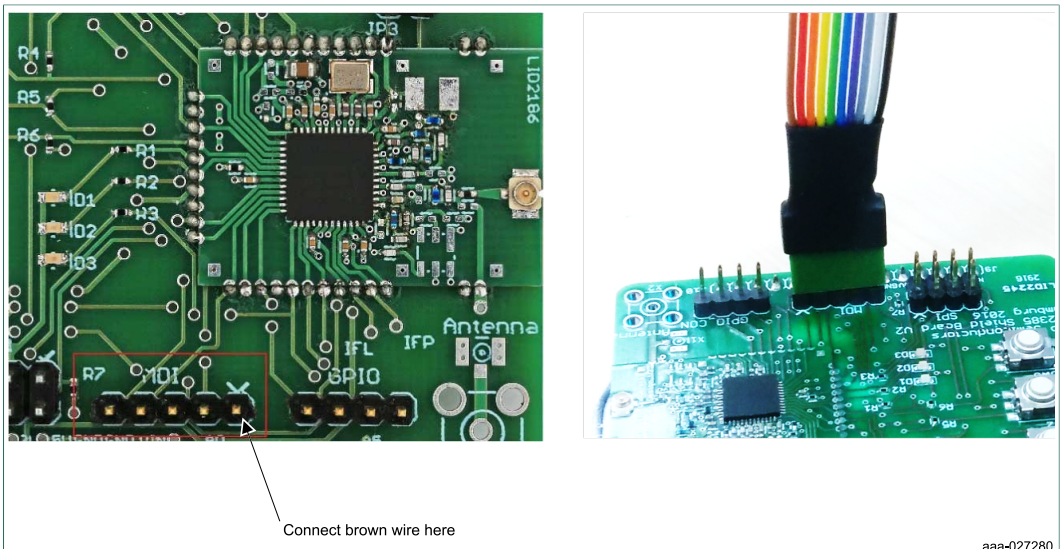


Figure 18. 2-link box connection

6 Setting up the software project

6.1 OL2385-SIGFOX firmware types

Depending on the interface to the OL2385 transceiver, the firmware running on it can be categorized into the following types:

- **SPI based firmware (fully available):** This version needs an external MCU connected to the OL2385 through a 5-wire SPI interface, as previously explained. This is a standard NXP product and readily available.
- **Button press based firmware (under investigation):** This version needs a small application to be written to the OL2385 to enable use of the three buttons provided on

the board. There is no external MCU required, thereby saving the customer additional cost. Because each application from customer to customer is different, this version will be provided as a skeleton with a combined library (NXP radio plus SIGFOX stack). The customer can modify the skeleton for customizing the behavior of the buttons and LEDs. A basic version of the software exists and can be enabled by a macro (BUTTON_PRESS_MODE) in project settings. The creation of a combined library still has to be done, therefore it is not ready to be provided. Additionally, the program memory does not have a lot of space and therefore, the application has to be very small and simple.

- **UART based firmware (under investigation):** This version needs a UART interface to the OL2385. This interface can also be to an external MCU or to a terminal program. From a business point of view, it makes sense to not have an external MCU required, thus saving the customer additional cost. A basic version of the software exists and can be enabled by a macro in project settings (SIGFOX_UART_MODE). However, the driver for UART is not robust and needs to be rewritten completely. The AT commands need to be updated and bugs have to be fixed.

Note: *A SIGFOX based module might go through a P1 certification process at SIGFOX, depending on if there is a hardware change from an NXP reference design or modification of certified NXP software. NXP has only standard product – SPI based firmware certified at SIGFOX for all regions. Even if the radio parts of the software remain unchanged, any small change in software will still be considered a change. And therefore, if a customer decides to use Button press based or UART based firmware, they will have to get their devices recertified. In order to do the certification, SIGFOX will require activation of various test sequences on the OL2385 device. This will be very difficult for Button press based firmware, because OL2385 has only three buttons. Therefore, a SPI or UART based interface to OL2385, where we can control these test sequences, is needed.*

Similarly, a UART based firmware without an MCU is good for evaluation using a terminal program. But for end product, customers will use either an MCU to connect to UART interface or use buttons again to write application on the device. In case of using an external MCU, it is highly recommended to use SPI based firmware, because it is certified and readily available. In case of no MCU usage and using buttons as a final interface to the user, UART will just be used as a certification interface. Again, in such a scenario it is easier to use a SPI based interface for certification purposes, because button applications will vary from customer to customer. It will again be our strategy to provide the code to customers with a combined library.

Proposed solution: For the applications where buttons are the interface to users and an interface is required only for evaluation and certification, it makes sense to provide the code to customers with a combined library and skeleton source files. A macro can be used by customers to enable/disable the SPI interface, which then will be used for one time certification. This means that customers do not have to use the MCU permanently on all products, but only externally to get certification. Therefore, there is no need for UART based firmware. Also, UART would have required an additional FTDI cable to interface OL2385 for certification. Now, it will just be an MCU based device controlling OL2385 externally for certification.

6.2 Kinetis MCU based application

This chapter is relevant for writing the code for an application on Kinetis MCUs. If your KL43Z is preflashed, you can skip to Section 7.1 "Testing on SIGFOX Base Station".

Otherwise, this chapter describes the MCU peripheral requirements and the resources needed to control a SIGFOX device.

6.2.1 Peripheral requirements

Peripherals and resource requirements critical to the MCU's ability to handle a given part are as follows:

- SPI module is required for communication (MISO, MOSI, SCK, CS)
- GPIO is required for an acknowledge pin (ACK)
- Supply and GND pins to power-on the OL2385

6.2.2 Supported devices

This section identifies the models supported by the driver and describes the capabilities of each model. The SIGFOX software driver supports the following devices:

OL2385 (Available)

- RF transceiver
- Carrier frequency is 160 MHz to 960 MHz
- ETSI, FCC and Japanese compliant
- 2 antenna inputs
- Independent RF switch (TX/RX or RX/RX)
- Up to +14 dBm output power
- 26 Channel Filter BW Options (4 kHz to 360 kHz)
- Smart polling
- Excellent phase noise
- Ultra-low power in receive mode

OL2361 (Software to be updated on demand)

- RF transmitter
- Carrier frequency is 310 to 915 MHz
- ETSI, FCC and Japanese compliant
- Multichannel fractional-N PLL
- One reference frequency (XTAL) for all bands
- Programmable FSK/ASK/OOK modulation characteristics
- Improved programmable and stabilized output power
- Low power consumption

6.2.3 Supported MCUs

The SIGFOX software driver supports MCUs listed in Table 3. These MCUs are a subset of the MCUs supported by the MCUXpresso Software Development Kit layer.

This software driver is built on the Analog Middleware Layer (AML), which creates an API abstraction layer for the desired Software Development Kit (SDK). The current implementation includes abstractions for MCUXpresso SDK 2.3 and S32 SDK 0.8.4 EAR. This allows support to be added for additional layers, such as the MCUXpresso SDK, without having to change the SIGFOX software driver itself.

Table 3. Supported MCUs in SDK

Supported MCUs	SDK
MKL43Z256 (FRDM-KL43Z)	MCUXpresso SDK 2.3
MKL27Z64 (FRDM-KL27Z)	MCUXpresso SDK 2.3
MKL25Z128 (FRDM-KL25Z)	MCUXpresso SDK 2.3
MK64FN1M0 (FRDM-K64F)	MCUXpresso SDK 2.3

See Table 4 for pin compatibility between the SIGFOX freedom board and selected MCUs.

Table 4. Compatibility of the SIGFOX board with selected MCUs

Pin function	FRDM-KL43Z	FRDM-KL27Z	FRDM-KL25Z	FRDM-K64F
ACK	PTD2	PTA5	PTD5	PTC4
CS	PTD4	PTC4	PTD0	PTD0
MOSI(SDI)	PTD6	PTC6	PTD2	PTD2
MISO(SDO)	PTD7	PTC7	PTD3	PTD3
SCK	PTD5	PTC5	PTD1	PTD1

6.2.4 The SIGFOX software driver

This section provides an overview of the functionality, configuration and usage of the driver. For additional information, see the API programmer's guide (included in the SIGFOX software driver zip file) and the comments in the code.

6.2.4.1 Configuring the driver

The configuration structure shown below serves as the user interface for configuring the driver. Users must add the appropriate values into the structure to reflect the desired driver setup. The configuration structure is passed directly to the initialization function at startup.

```
/*! @brief This structure is used by the user to initialize the SIGFOX device.
 * It contains a configuration of the SIGFOX device only (no SPI,
 * etc. configuration).  */
typedef struct
{
   sf_net_standard_t netStandard;  /*!< Selection of a standard defining network
                                        communication parameters. */
   uint8_t txAttSteps;             /*!< Amount of 0.25 dB steps that needs to be
                                        subtracted from TX power. */
   sf_xtal_type_t xtal;            /*!< Selection of XTAL type used in the
                                        device. */
   sf_tx_repeat_t txRepeat;        /*!< Amount of transmissions that will happen
                                        on one time invoking of SF_SendPayload
                                        function. */
   sf_pa_type_t paType;            /*!< Selection of internal PA type used in
                                        the device. */
} sf_user_config_t;
```

The user's configuration structure includes the following variables:

- **netStandard** selects a standard for defining network communication parameters
- **txAttSteps** contains number of 0.25 dB steps that needs to be subtracted from the TX power
- **xtal** selects between usage of 55.2 MHz and 27.6 MHz TCXO XTAL
- **paType** selects between internal PA being used: 14 dBm or 0 dBm
- **txRepeat** selects number of transmissions that will happen when invoking SF_SendPayload function (1TX or 3TX).

For a more detailed description of the user configuration structure, refer to the API programmer's guide included in the SIGFOX software driver zip file.

6.2.4.2 Driver API

This software driver provides an API that allows user application code to configure the device in real time. For a summary of the available functions, see Table 4.

Table 5. SIGFOX software driver API

Function	Description
SF_Init	Initializes the SIGFOX driver based on user configuration
SF_GetDefaultConfig	Loads the user configuration structure with default values
SF_SendCommand	Sends a command to the device. This function waits for an acknowledgment (if any)
SF_WakeUp	Sends the **Send_Wakeup** command to wake up the device from power down mode
SF_Sleep	Puts the device in power off mode by means of the **Send_To_Sleep** command
SF_SetUlFrequency	Sets the uplink frequency of the SIGFOX device
SF_GetUlFrequency	Gets the uplink frequency of the SIGFOX device
SF_SendPayload	Sends the **Send_Payload** command to the device, which sends user data to SIGFOX network
SF_SendBit	Sends the **Send_Bit** command, which transmits a single bit to the SIGFOX network. This is the shortest frame that the SIGFOX library can generate. This command is useful for network testing.
SF_SendOutOfBand	Sends the **Send_out_of_band** frame, which transmits data to the SIGFOX network. Data is composed of information about the chip itself (voltage, temperature). This function must either be called at least once every 24 hours or never, depending on whether an application has some energy critical constraints.
SF_ReceiveMessage	Receives a frame from SIGFOX network using the **Receive_Frame** command. Available for the OL2385 device only
SF_TriggerWatchdog	Resets the SIGFOX device watchdog using the **Trigger_Watchdog** command.
SF_GetDeviceInfo	Reads Device ID, PAC, SIGFOX library version and device version using the **Get_info** and **Get_Device_Version** commands
SF_GetState	Returns the current state of the SIGFOX device in the software state machine. The state is updated every time an ACK SPI frame is received.
SF_GetErrorCode	Returns an error code received in a SPI frame from the SIGFOX device. The error code is updated every time an ACK SPI frame is received.
SF_TestSpiCon	Tests if the SPI bus is working by using the **Echo** command to send data (see SF_ECHO_DATA macro) and then checking if the device replies with the inverted payload
SF_TestDevice	Executes a test of uplink and downlink connectivity using the **Send_Test_Mode** command. The returned status indicates the success or failure of the test. The user can obtain details using the **SF_Get_Error_Code** function.
SF_GenerateContWave	Sends the **Continuous_Wave** command to generate a continuous transmission at the last set frequency. The signal can be then analyzed using a spectrum analyzer.

Function	Description
SF_ChangeNetworkStandard	Changes the up-link and down-link frequency and bit rate, depending on which of the following standards is enabled: European ETSI standard (default setting), USA FCC standard, Japanese/ Korean ARIB standard and South American FCC standard. Uses **Change_To_RCZ1** (European ETSI), **Change_To_RCZ2** (USA FCC),**Change To RCZ3** (Japanese/Korean ARIB) and **Change_To_RCZ4** (South American FCC) to set the appropriate standard.
SF_SwitchToPrivateKey	Sends the **Switch_to_Private_Key** command in order to switch to the private key provided by SIGFOX. It is used for normal RF transmission.
SF_SwitchToPublicKey	Sends the **Switch_to_Public_Key** command in order to switch to the public key provided by SIGFOX. It is used for protocol tests using SFXIQ or SNEK emulator.
SF_TransmitZeroSeq	Sends modulated zeros at currently set frequency using **PSK_ASK_Zero_Sequence** command.
SF_TransmitAAAASeq	Sends modulated 0xAA at currently set frequency using **PSK_ASK_AAAA_Sequence** command.
SF_SetDlFrequency	Sets down-link frequency of the SIGFOX device via **Set_DL_Frequency** command.
SF_GetDlFrequency	Gets down-link frequency of the SIGFOX device via **Get_DL_Frequency** command.
SF_SetFCCMacroChannel	Sets the FCC macro channel by sending the **Set_FCC_Macro_Channel** command.
SF_GetFCCMacroChUsage	Gets the FCC macro channel usage by sending the **Get_FCC_Macro_Channel_Usage** command.
SF_ResetFCCMacroChannel	Restores the default macro channel settings using **Reset_FCC_Macro_Channel** command.
SF_CalibrateFreqStatic	Calibrates TX frequency using **Static_Frequency_Calibration** command. The passed drift offset measured in Hz is added or subtracted to the LO when doing RF TX.
SF_UpdateTempFreqCalTable	Updates temperature frequency calibration table using the **Temperature_Frequency_Calibration** command.
SF_ReadTempFreqCalTable	Reads temperature frequency calibration table using the **Temperature_Frequency_Calibration** command.
SF_SetDefaultTempFreqCalTable	Sets default temperature frequency calibration table using the **Temperature_Frequency_Calibration** command.
SF_GetLastSentRSSI	Gets the recently computed RSSI using the **Get_Last_Sent_RSSI** command.
SF_GetLast30SentRSSI	Sends the **Get_Last_Sent_RSSI** command to get the recently computed RSSI measure for 30 frames during RX GFSK test. This is the receiver sensitivity test.
SF_ResetLastSentRSSIBuffer	Sends the **Get_Last_Sent_RSSI** command to reset the buffer, which holds the recently computed RSSI for 30 frames during RX GFSK test. This is the receiver sensitivity test.
SF_ReadCurrentPATable	Reads current ETSI/FCC PA table using the **Update_PA_Curve** command.
SF_UpdatePATable	Updates the ETSI/FCC PA table using the **Update_PA_Curve** command.
SF_SetDefaultPATable	Sets default ETSI/FCC PA table using the **Update_PA_Curve** command.
SF_CalibrateInitialRSSI	Sets the initial RSSI drift offset using the **Initial_RSSI_Calibration** command.
SF_SendCommandNonBlock	Sends a command to the device, but does not wait for an acknowledgment. Use the **SF_HAS_CMD_ACK** macro to check if the selected command has issued an ACK. Then use the **SF_Is_Ack_Frame_Ready** and **SF_Read_Ack_Frame_NonBlock** functions to receive the ACK.
SF_IsAckFrameReady	Checks if the SIGFOX device is ready to send an acknowledgment. It checks the ACK pin value, which is active low. The user can use the **SF_HAS_CMD_ACK** macro to check if a command has issued an acknowledgment.
SF_ReadAckFrameNonBlock	Receives an acknowledge frame. Should be used in conjunction with the **SF_Send_Command_NonBlock** and **SF_Is_Ack_Frame_Ready** functions. Use the **SF_HAS_CMD_ACK** macro to check if a command has an acknowledgment.
Peripherals setup	The following functions are placed in the sf_setup.h header file.
SF_SetupSPI	Configures a SPI module used by the SIGFOX device.
SF_SetupGPIOs	Configures the GPIOs used for the SIGFOX device. Note that this function should not be used in SDK S32 when using the pin_mux component.

For a more detailed description of the software driver API (function signatures, parameters) refer to the API programmer's guide (included in the SIGFOX software driver zip file).

6.2.4.3 Blocking and nonblocking functions

The SIGFOX driver includes functions for blocking and Nonblocking SPI communication. Blocking functions send a SPI frame and wait for an acknowledgment frame from the sample driver. Nonblocking functions send a SPI frame and immediately return. The user has to call another function to receive an acknowledgment.

For the most part, the SIGFOX driver uses blocking communication functions because they are simpler to implement. The user only needs to call a blocking function to send a SPI command and to get a response containing the desired data.

Nonblocking functions are useful when the wait time for a response is several seconds. This applies to SPI commands that involve transmissions to the SIGFOX network, such as Send Payload, Send Bit and Send Out of Band. Nonblocking communication for these types of SPI commands is more practical because the MCU can do useful work while waiting for the acknowledgment. The functions in Table 6 implement nonblocking communication, Figure 19 shows typical usage of these functions.

Table 6. Nonblocking functions

Function name	Description
SF_SendCommandNonBlock	Sends a SPI command to the SIGFOX device and does not wait for an acknowledgment
SF_IsAckFrameReady	Checks if the SIGFOX device is ready to send an acknowledgment
SF_ReadAckFrameNonBlock	Reads an acknowledgment frame

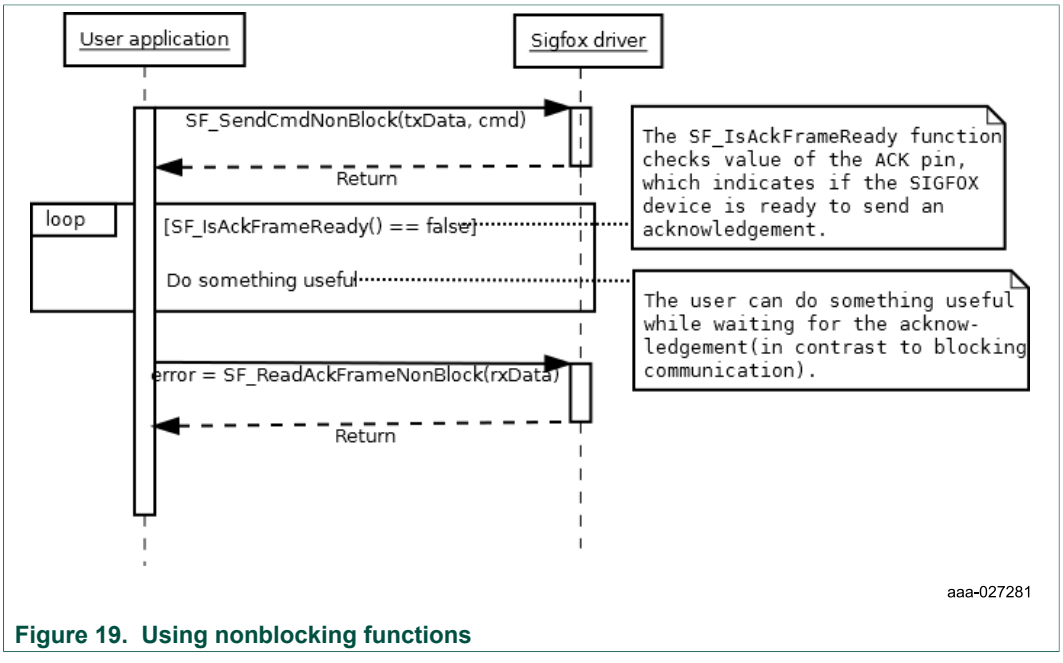


Figure 19. Using nonblocking functions

6.2.4.4 SPI command descriptions

For details of each command, refer to SIGFOX SPI Command Interface Description.xls at:

www.nxp.com/products/software-and-tools/hardware-development-tools/freedom-development-boards/om2385-sf001-ol2385-wireless-sub-ghz-transceiver-sigfox-development-kit-with-kl43z:OM2385-SF001?tab=Documentation_Tab

6.2.4.5 Low-level drivers

This section describes the low-level MCU peripheral drivers used by the SIGFOX software driver. The SIGFOX driver depends on these low-level drivers for functions such as communication and control.

Because the SIGFOX software driver is built on AML, it has access to a subset of low-level drivers supported by the underlying SDK. Most analog devices require that some of the SPI, I^2C, Timer, ADC and GPIO drivers be configured on the MCU side. The AML layer unifies the API for these selected drivers, making the software driver portable across SDKs.

Figure 20 illustrates the software driver architecture in terms of communication and control.

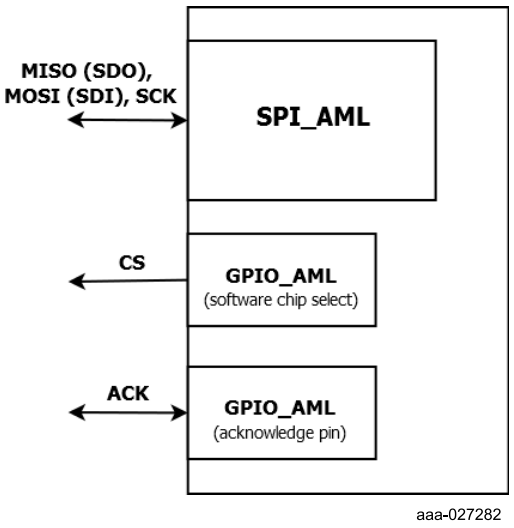


Figure 20. Software driver architecture

6.2.4.6 Required driver setup

In order to execute correctly, the SIGFOX software driver requires the following:

- In the file aml/common_aml.h, the SDK_VERSION macro must indicate which SDK is being used.
- In the file sf/sf_model.h, the SF_MODEL macro must indicate which SIGFOX model is being used. Setting this parameter enables the driver to make the necessary adjustments required to support the model.

6.2.5 Installing the software using KDS

This section describes how to install Kinetis Design Studio and use the embedded software driver for SIGFOX application development.

MCUXpresso SDK is a comprehensive collection of software resources that supports NXP microcontroller development. For more information, go to the website mcuxpresso.nxp.com.

6.2.5.1 Installing Kinetis Design Studio

This procedure explains how to obtain and install the latest version of Kinetis Design Studio (version 3.2.0 in this guide). The procedure for MCUXpresso IDE installation or S32 Design Studio installation is similar.

Note:

The component and some examples in the component package are intended for Kinetis Design Studio 3.2.0 or newer and MCUXpresso IDE 10.1.1 or newer. If Kinetis Design Studio 3.2.0 or MCUXpresso 10.1.1 is already installed on the system, skip this section.

1. Obtain the latest Kinetis Design Studio 3.2.0 installer file from the NXP website here: www.nxp.com/KDS
2. Obtain the MCUXpresso IDE installer file from NXP website here: www.nxp.com/MCUXpresso
3. Obtain the S32 Design Studio installer file from the NXP website here: www.nxp.com/S32DS

Run the executable file and follow the instructions.

6.2.5.2 Downloading the MCUXpresso SDK library

This step is necessary only when creating a new MCUXpresso SDK project, in which case the selected MCU's low-level drivers must be downloaded. The example projects listed in Table 7 include the required drivers. In addition, the S32 Design Studio SDK library is distributed with a complete IDE and requires no manual driver additions.

The MCUXpresso SDK Builder is a tool that allows the downloading of a customized MCUXpresso SDK based on a specific evaluation platform or Kinetis MCU. To download the MCUXpresso SDK, go to: mcuxpresso.nxp.com

6.2.5.3 Downloading the software driver and example projects

To download the latest version of the SIGFOX software driver and the example projects, do the following:

1. Go to the NXP website www.nxp.com/EMBEDDED-SW-SIGFOX.
2. Download the zip file containing the software driver and the example projects.
3. Unzip the downloaded file and check to see that the folder contains the files listed in Table 7.

Table 7. Content of downloaded zip file

Folder name	Folder content
KDS_Examples	Example project folder for Kinetis Design Studio 3.2.0 or newer.
FRDM_K64F_OL2385_ConsoleControl	FRDM-K64F example project that allows users to control the SIGFOX device via a virtual serial port.
FRDM_KL27Z_OL2385_ConsoleControl	FRDM-KL27Z example project that allows users to control the SIGFOX device via a virtual serial port.
FRDM_KL43Z_OL2385_ConsoleControl	FRDM-KL43Z example project that allows users to control the SIGFOX device via a virtual serial port.
FRDM_KL43Z_OL2385_Demo	KL43Z example project that collects data from sensors placed on the MCU board and sends them to the SIGFOX network. The second part is an HTML application that shows data from the SIGFOX server.
FRDM_KL43Z_OL2385_DemoExtSensors	KL43Z example project that collects data from MCU sensors and external sensors (GPS, pressure sensor) and sends them to SIGFOX network. The second part is an HTML application that shows data from the SIGFOX server.
MCUXpresso_Examples	Example project folder for MCUXpresso IDE v10.1.1 or newer.
FRDM_K64F_OL2385_ConsoleControl	FRDM-K64F example project that allows users to control the SIGFOX device via a virtual serial port.
FRDM_KL27Z_OL2385_ConsoleControl	FRDM-KL27Z example project that allows users to control the SIGFOX device via a virtual serial port.
FRDM_KL43Z_OL2385_ConsoleControl	FRDM-KL43Z example project that allows users to control the SIGFOX device via a virtual serial port.
Programmer_Guide	This folder contains API documentation generated directly from the code. (The API programmer's guide is not available online.)
SDK_SW_Driver	SIGFOX Software Driver folder. This folder contains two subfolders: **aml** and **sf**.

6.2.5.4 Importing an example project into Kinetis Design Studio

The following steps show how to import an example from the downloaded zip file into Kinetis Design Studio. In MCUXpresso IDE, the following steps are similar.

1. In the Kinetis Design Studio menu bar, click **File** → **Import…** In the pop-up window, select **General** → **Existing Projects into Workspace** and click **Next**.
2. Click Browse and locate the folder containing the unzipped example files. Find the folder: **Sigfox_SDK_SW\KDS_Examples** and select a project to import. Then, click **OK**.
3. With the project now loaded in the **Select root directory** box, click on the **Copy projects into workspace** check box. Then click **Finish**. The project is now in the Kinetis Design Studio workspace where it can be built and run.

6.2.5.5 Creating a new project with the SIGFOX software driver

For users electing not to use one of the provided example projects, the following instructions describe how to create and set up a new project that uses the SIGFOX software driver.

To create a new project, do the following:

1. In the Kinetis Design Studio menu bar, select **File** → **New** → **Kinetis SDK 2.x Project**. When the Select Kinetis SDK 2.x box opens, enter a project name into the text box. Select the path to the SDK 2.x library and then click **Next**. Note that the SDK package for the relevant MCU must have been previously downloaded. See Section 6.2.5.2 "Downloading the MCUXpresso SDK library".

2. In the **Select Kinetis Processor or Board** dialog box, select the MCU class or board the project is using. Then, click **Next**.
3. A new project is created. All of the required files are placed in dedicated folders: **board**, **CMSIS**, **drivers**, **source**, **startup** and **utilities**. For a more detailed description of the SDK project folder structure, refer to the MCUXpresso 2.x User Guide.

6.2.5.6 Adding the SIGFOX software driver to the project

This section describes how to add the SIGFOX Software driver to the project.

1. Copy the content of **Sigfox_SDK_SW\SDK_SW_Driver** to the source folder in the newly created project.
2. In main.c, add the include statement highlighted in the figure below to enable the code to access the SIGFOX software driver.

6.2.5.7 Setting up the project

Once the new project has been created and the SIGFOX software driver has been added into it, the project must be set up. See Section 6.2.4 "The SIGFOX software driver", which describes the driver's capabilities and what must be done to configure its properties.

1. In order to get the SIGFOX software driver to run on the selected MCU, the user must edit the board/pin_mux.c file to configure the SPI and GPIO pins being used. This entails making the correct MCU pin selections (see Section 6.2.3 "Supported MCUs") and then muxing them as needed. Use one of the example projects as a template for this step. See the figure below.
2. In the file aml/common_aml.h, set the SDK_VERSION macro to SDK_KSDK (MCUXpresso SDK 2.3).
3. In sf/sf_model.h, update the SF_MODEL macro according to the SIGFOX model being used.
4. Create a variable of type sf_drv_data_t that will be passed to all used functions. This variable stores MCU peripheral configuration data and SIGFOX software driver internal data. This variable must be accessible during run-time and should be declared either in the main() function or as a global variable.
5. Configure the SPI and pin settings in the driver configuration structure.
6. Set up the MCU peripherals that will be used by the SIGFOX software driver. There are two options:
   - Use the provided SF_SetupSPI() and SF_SetupGPIOs() functions, which will do the necessary configuration.
   - Code the peripheral initialization without using the provided functions.

   In either case, the configuration structure must be modified to indicate the instances of utilized peripherals.
7. Call the SIGFOX initialization function. The initialization function must be passed the references to the already configured MCU peripherals, which is part of the driver data structure, and the user configuration of the device itself.

6.2.5.8 Writing application code

All application code must reside in the source folder in the project directory. The code in main.c may be modified but the original comments related to usage directions should be retained.

When the SIGFOX software driver is configured properly, the user can take advantage of all of the prepared functions to construct the application. See the API Programmer's Guide, included in the SIGFOX software driver zip file, for function signatures and required parameters. Also, review the sf/sf.h header file, which contains prototypes for all available functions.

For more complex driver use cases, refer to the included example projects.

6.2.5.9 Compiling, downloading and debugging

To compile a project, click the compile icon in the toolbar.

The process for downloading an application onto an MCU board in Kinetis Design Studio differs according to the type of board. For more information, see the Kinetis Design Studio V3.2.0-User's Guide. To download and debug on the KL43Z MCU board, do the following:

1. Click the arrow next to the debug icon in the toolbar and select **Debug Configurations…**
2. In the **Debug Configurations** dialog box, click **KL43_SDK_Project_Debug_PNE** under **GDB PEMicro Interface Debugging**.
3. Click the **Debugger** tab and set the Interface option to OpenSDA Embedded Debug – USB Port. Then click the **Refresh** button next to the **Port setting** to update the list of available USB ports.
4. Then make sure that the Target is set to KL43Z256M4. If not, click on the **Select Device** button. In the **Select Target Device** dialog box go to **NXP → KL4x → KL43Z256M4**. Confirm the selection by clicking the **Select** button.
5. Click **Debug**. Kinetis Design Studio will download and launch the program on board.

7 Testing the SIGFOX application

7.1 Testing on SIGFOX Base Station

SIGFOX backend can be used for a basic functionality check of the SIGFOX module. This includes reception of a SIGFOX message and display of its payload along with repetition number. The base station can be leased from SIGFOX or generally is deployed by SIGFOX in your city. Visit www.SIGFOX.com/en/coverage to check coverage in your city. The base station is connected to the cloud by the internet; and it can receive a SIGFOX message sent by an OL2385 board and display it on a web browser.

7.1.1 Registering SIGFOX device on network

1. Take a note of the Device ID and PAC of the device. If the device is preflashed, these numbers are already flashed inside. If you have SPI based firmware on OL2385 (check the section Section 5.4.1 "Setting up KL43Z board with terminal program"), you can get the Device ID and PAC using the SPI command Get info. For UART based firmware, send the command AT$ID? via an FTDI cable. If the device is not preflashed, contact NXP customer support.
2. Activate your device by registering it on backend.SIGFOX.com/activate.
3. Choose your kit provider from the list as shown in the figure below.
4. Fill in the subscription information as shown below and click Subscribe.
5. Check your email for a message regarding your account.

6. Set a username and password to complete the account creation.
7. Go to backend.SIGFOX.com/auth/login to log into your account, as shown in Figure 23.
8. Click on the **Device** tab to view your registered devices.
9. If you see your device with your registered device ID, the registration is successful. The device and signal information will be updated on reception of first SIGFOX message sent by the device and received by any SIGFOX base station.
10. If you do not find your device, contact support@SIGFOX.com with your device ID and the PAC value you tried to use for device registration. Inform them about your country of operation and your chip supplier, NXP.

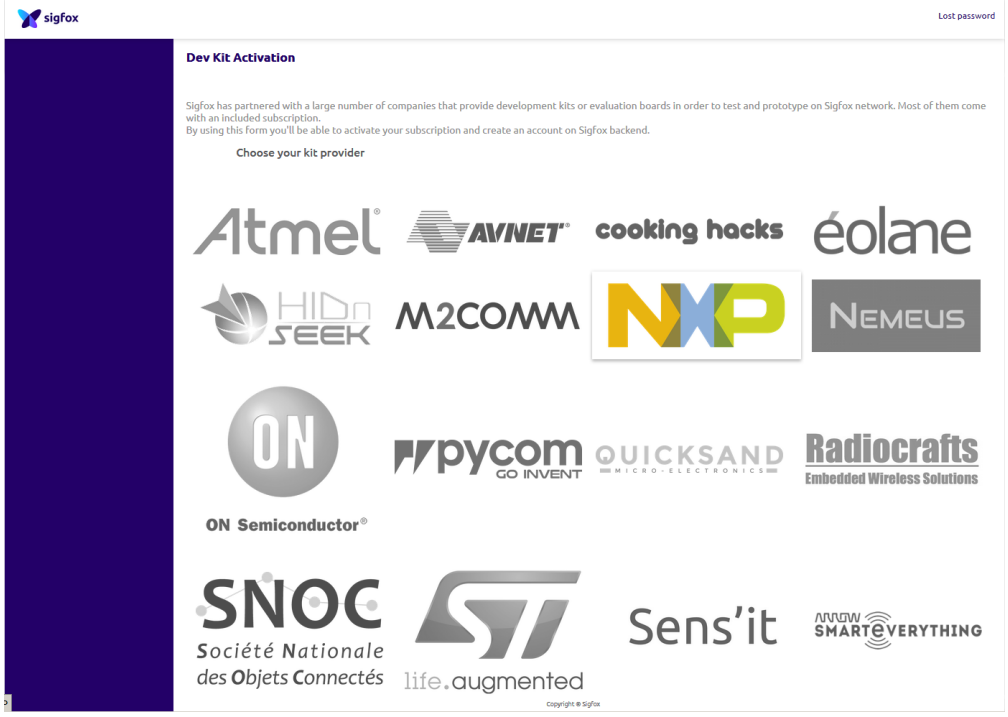


Figure 21. Kit providers

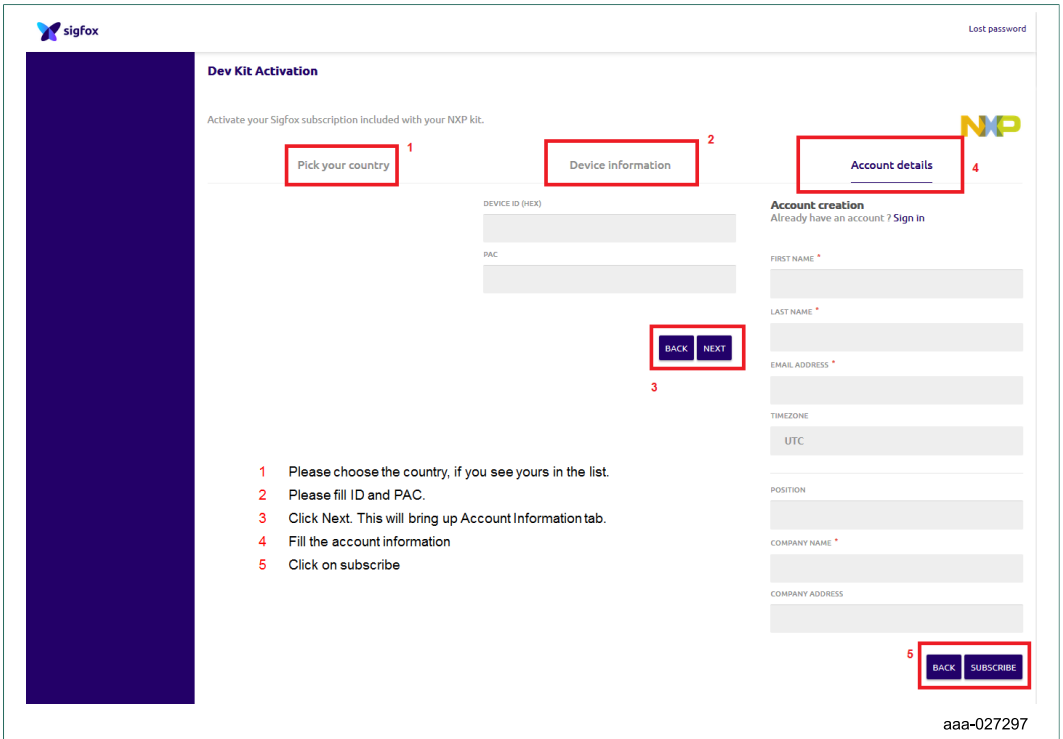


Figure 22. Registration form on SIGFOX backend

Note: *The device ID belongs to the following four regions in which SIGFOX has divided the world for its service:*

- *RCZ1 ETSI Europe (863 to 870 MHz)*
- *RCZ2 FCC US (902 to 928 MHz)*
- *RCZ3 ARIB Japan, Korea (915 to 930 MHz)*
- *RCZ4 FCC Latin America, Australia, New Zealand (902 to 915 MHz)*

It is possible that the ID-PAC combination you received is valid for a different region. Due to this, registration might not be successful. However, contacting suport@SIGFOX.com will provide a solution, because they can change the area of the device operation at the backend.

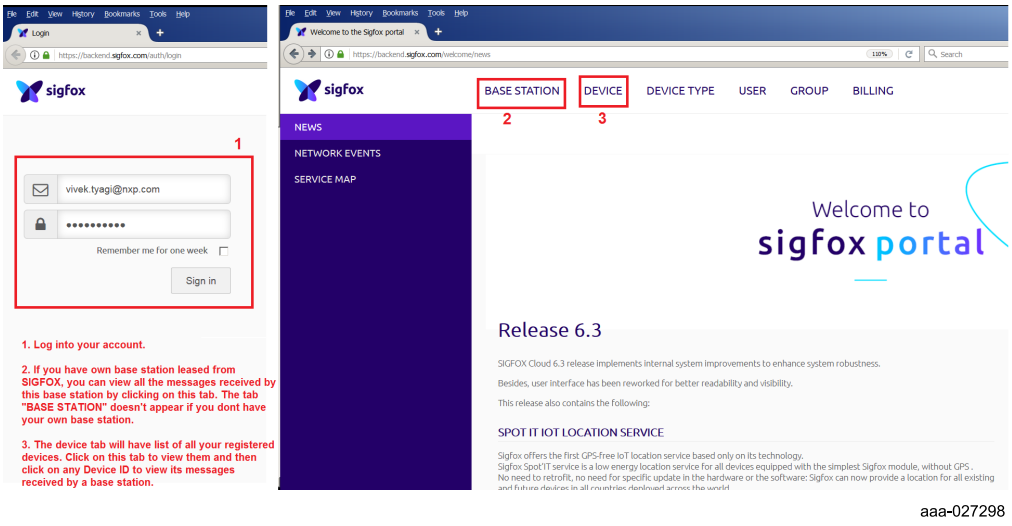


Figure 23. Logging into your SIGFOX account

7.1.2 Verifying the successful transmission

After registering the device at the backend, the device can now transmit a SIGFOX message that can be viewed on a browser.

Note: *If you are not using your own leased base station from SIGFOX, you will not be able to see the BASE STATION tab on the portal, which you can see in these sample figures. In that case, you can only see the rest of the tabs.*

1. Log into your SIGFOX account by visiting: backend.SIGFOX.com/auth/login.
2. Click on the **DEVICE** tab to see all of your registered devices, as shown in Figure 24.
3. Click on the device ID of the device for which you would like to verify transmission.
4. Make sure you have the right firmware on the device. Check Section 6.1 and Section 5.4.2 in case you have to update the device firmware.
5. Prepare the hardware connections of your device as per Section 5.4.1.2 "Connecting the hardware for use with the SIGFOX network". Power-on your device.
6. Depending on the choice of firmware, send a SIGFOX message on radio by performing any of the following actions
   a. Send SPI command **Send payload**. See Section 5.4.1 "Setting up KL43Z board with terminal program" for sending SPI commands. If the hardware is freshly powered on or reset, send the sequence: 1 (Send wakeup), 15/16/17/18 (Change_To_RCZX depending on the region), 4 (Send payload)
   b. Or do a button press (If button press based firmware on device)
   c. Or send an AT command via terminal on PC to send a SIGFOX frame (If UART based firmware on device)
7. Click on the **Messages** tab on the portal as shown in Figure 25.
8. Depending on the contract you have for your device with SIGFOX, you are able to see certain fields for each message. This is dependent on the amount of access rights you have for your device. For more details, contact support@SIGFOX.com. A minimum of following fields should at least be visible: Time, Data / Decoding, Location, Link quality, Callbacks. An example of extended fields with higher access is shown in the figure below.
9. Similarly, the messages can be sent by the device and checked here at the portal any number of times. Generally speaking, one can see the device message appears on

the portal just after a few seconds of actual transmission. If the message does not appear within a minute, it is lost. Check your region settings on the terminal and try again.

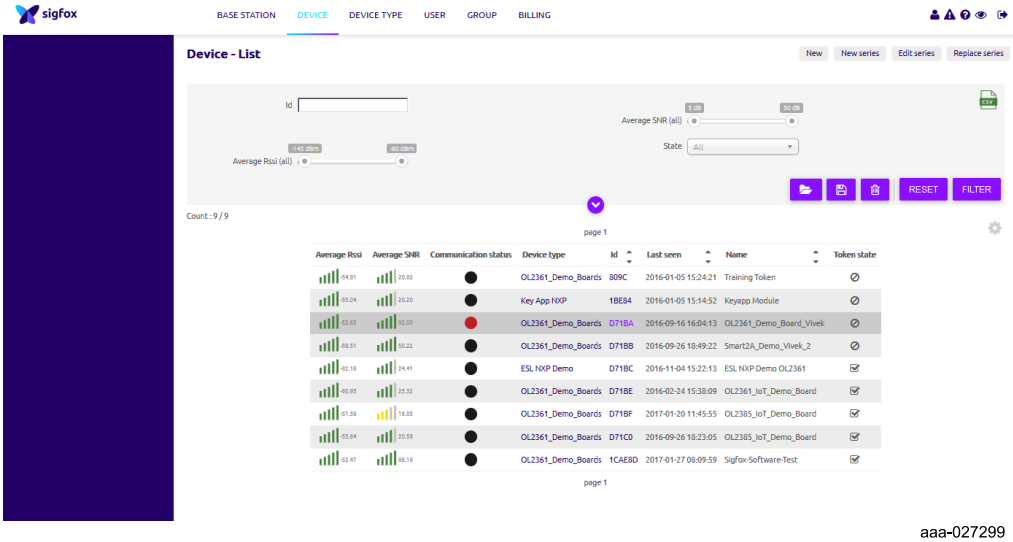


Figure 24. List of all registered devices on backend

Note: *Check the middle LED on the NXP reference design board. If not specifically programmed for another purpose, its blinking indicates the transmission of frames. Each call of Send payload is, by default, configured for sending a frame on three different frequencies (frequency hopping). Therefore, one command of sending a frame on terminal or button press results in the middle LED blinking three times.*

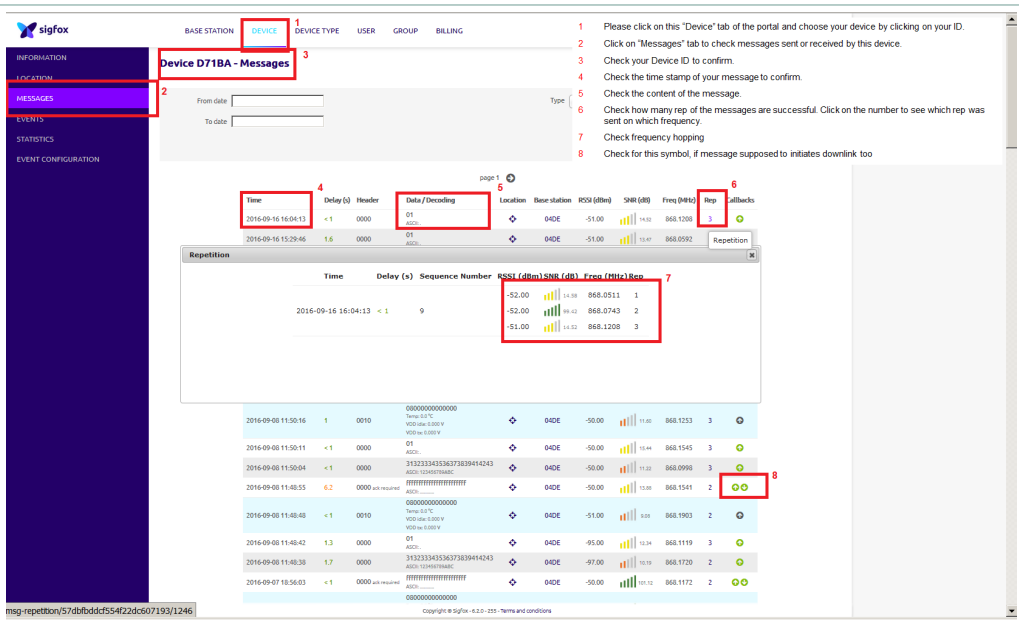


Figure 25. Checking successful SIGFOX message transmission on backend

7.2 OL2385 Transceiver Initialization Sequence

After you have set up the KL43Z board according to Section 5.4.1 "Setting up KL43Z board with terminal program", you are now ready to use the terminal to control the OL2385 board. In order to do so, the following sections have to be understood.

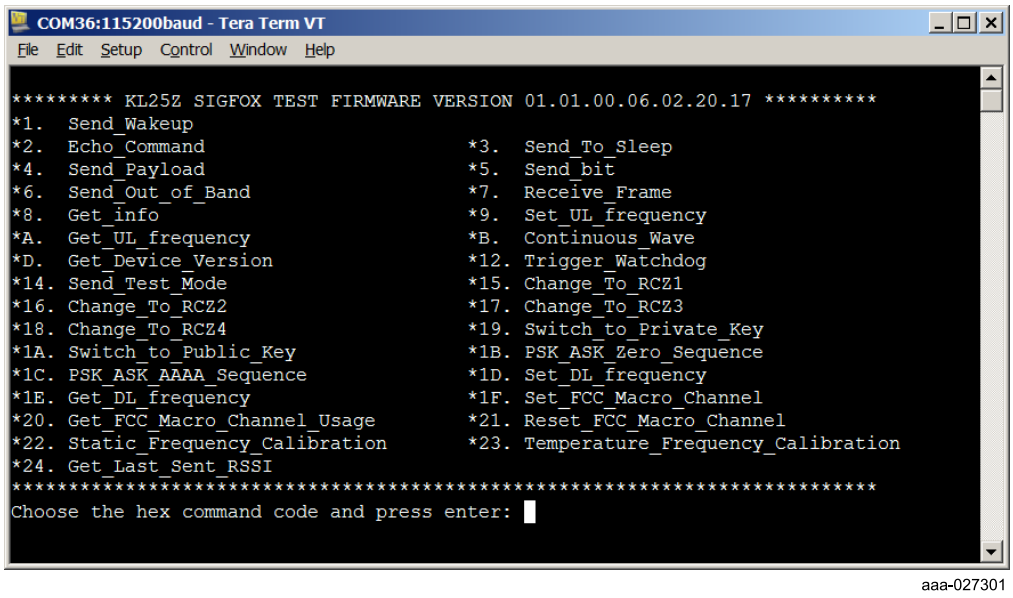


Figure 26. SPI commands to control OL2385

7.2.1 Mandatory initialization sequence

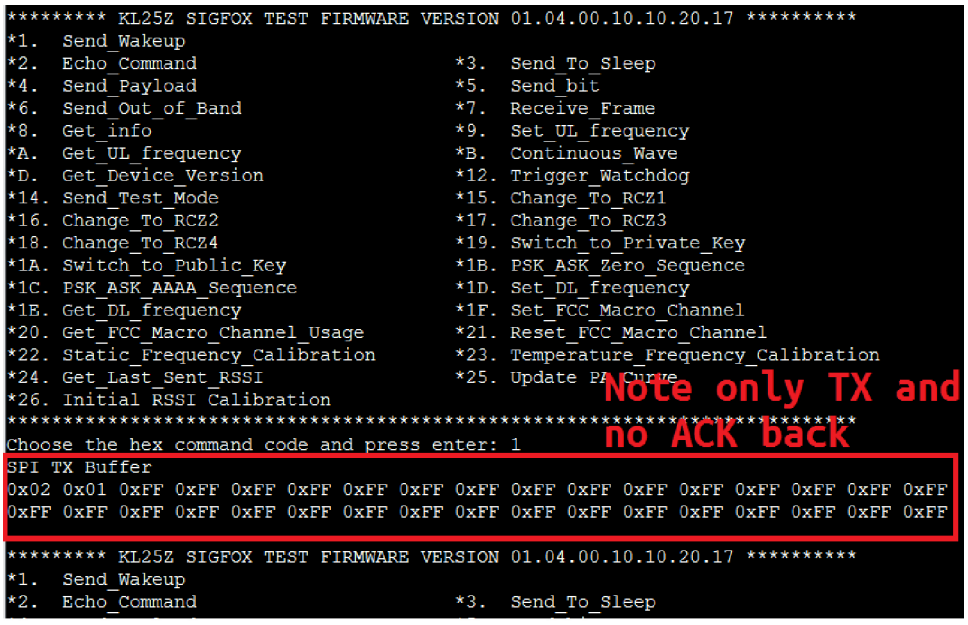
The following sequence of commands is a **mandatory** initialization sequence that must be executed after power-on/reset of the OL2385 board.

7.2.1.1 Wakeup from low-power mode

A Send wakeup command has to be sent to the device first after any kind of software device reset, power on/off reset or battery-on reset. By default, the device goes immediately to low-power SLEEP mode after any of these resets or after the explicit SPI command Send to sleep. The CS pin of the SPI interface is the wakeup pin and has to be driven low by the MCU to wake the device up. This is done by a Send wakeup command, including two bytes transfer as per SPI protocol. This has to be done only once after the reset until the next reset. Once done until the next reset, other commands can be used.

Note: *Before proceeding to the low-power mode, OL2385 configures all of its pins to a pulled-down configuration, except for CS and ACK pins of SPI—these pins are pulled high. For the low-power mode to work properly without any leakage current, it is important to have the same settings of the pins on the MCU side. All pins on the MCU should be pulled low and only CS and ACK should be high.*

Note: *The Send Wakeup command does not return an acknowledgment RX frame back.*



aaa-029131

Figure 27. Wakeup from sleep SPI command

7.2.1.2 Choosing the right region

The next command has to be about setting the region "Change to RCZXX". This command is needed one time only after reset until the next reset. SIGFOX has divided the world for its service into four regions:

- RCZ1 ETSI Europe (863 to 870 MHz)
- RCZ2 FCC US (902 to 928 MHz)
- RCZ3 ARIB Japan, Korea (915 to 930 MHz)
- RCZ4 FCC Latin America, Australia, New Zealand (902 to 915 MHz)

These regions decide the uplink-downlink frequency and the data rate. The choice of the region also decides the hardware used for the device. RCZ2/4 devices have an external PA on themselves whereas an RCZ3 device has an external TCXO(27.6MHz) used instead of 55.2 MHz normal crystal. Therefore, software will only function properly, if the right region is chosen according to correct hardware. Therefore, "Change to RCZXX" should be the next command.

Note: *By default, device wakes up in RCZ1 mode*

Table 8. Regional center frequencies

By default, device wakes up in RCZ1 mode

Region	Opening Uplink center frequency (MHz)		Opening Downlink center frequency (MHz)	Bandwidth (kHz)
RCZ1	868.13		869.525	192 (±96)
RCZ2	902.2		905.2	192 (±96)
RCZ3	923.2		922.2	36 (±18) evolution to 192 (± 96)
RCZ4	**Opening UL center freq.**	**TX freq. used in SFXIQ testing**	922.3	192 (±96)
	902.2	920.8		

Note: *For any RCZ, there can be a difference between center frequency in the specification and opening center frequency for the SIGFOX library due to the frequency-hopping algorithm. For example, the actual TX frequency for RCZ4 is 920.8 MHz but the value in the previous table is 902.2 MHz. This is because 902.2 MHz is the opening/ starting value of frequency in the frequency map. The frequency map is actually used to enable all the possible frequencies on which the device can hop for transmitting frames. The channels in this map are enabled using config word settings. See the SIGFOX_api.h file for detailed information. So, for RCZ4 the map will start enabling the first frequency starting from 920.8 MHz, which means that the frequency-hopping algorithm will automatically take care of the correct frequency value. All this algorithm needs are the initial opening values as input, from which it calculates the right frequency. Therefore, if you will just use continuous wave after changing to RCZ4 region and check on spectrum analyzer you will see only the 902.2MHz, since FH algorithm is not considered in CW. However, when you will do a frame transmission using "Send payload" SPI command or test mode sequence "14-0-9" to send 9 frames on constant frequency, the frequency of transmission will be 920.8 MHz.*

Additionally, the Change Zone commands allows the user to configure the following:

- Number of attenuation steps: 1 step is equal to 0.25 dBm of attenuation. *The default value is 0.*
- Board oscillator type: 0 for standard crystal oscillator (XTAL) and 1 for temperature calibrated crystal oscillator (TCXO).

Note: *NXP currently only supports boards operating a TCXO and not a standard XTAL oscillator.*

- SIGFOX frame repetitions: As per SIGFOX library version 1.9.2a, which is included in OL2385 firmware version 01.12.00.02.08.20.17, a value of 0 will set it to 1 frame. Any value more than zero will set it to 3 repetitions. *The default value is 3.*
- Internal PA type: 0 for 14 dBm internal PA or 1 for 0 dBm internal PA. *The default value is 0 (14d Bm internal PA)*.

How to use the Change Zone command:

1. After wakeup from sleep, type **15** for RCZ1, **16** for RCZ2, **17** for RCZ3, or **18** for RCZ4, and then press Enter.
2. Type the number of TX attenuation steps, such as 0, and then press Enter.
3. Type **0** if the board operates on a standard crystal oscillator (XTAL) or type **1** if the board operates on a temperature calibrated crystal oscillator (TCXO), and then press Enter.
4. Type the number of SIGFOX frame repetitions, such as 3, and then press Enter.
5. Type **0** for 14 dBm internal PA or type **1** for dBm internal PA (*default value is 0*), and then press Enter.

Macro Channel Value MHz :	902.2MHz	902.5MHz	902.8MHz	903.1MHz	903.4MHz	903.7MHz	904.0MHz	904.3MHz	904.6MHz	904.9MHz	905.2MHz	... ...	911.5MHz
Macro Channel Value :	Chn 1	Chn 2	Chn 3	Chn 4	Chn 5	Chn 6	Chn 7	Chn 8	Chn 9	Chn 10	Chn 11	... ...	Chn 32
config_words[0] bit :	bit 0	bit 1	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7	bit 8	bit 9	bit 10	... ...	bit 31
Macro Channel Value MHz :	911.8MHz	912.1MHz	912.4MHz	912.7MHz	913.0MHz	913.3MHz	913.6MHz	913.9MHz	914.2MHz	914.5MHz	914.8MHz	... ...	921.1MHz
Macro Channel Value :	Chn 33	Chn 34	Chn 35	Chn 36	Chn 37	Chn 38	Chn 39	Chn 40	Chn 41	Chn 42	Chn 43	... ...	Chn 64
config_words[1] bit :	bit 0	bit 1	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7	bit 8	bit 9	bit 10	... ...	bit 31
Macro Channel Value MHz :	921.4MHz	921.7MHz	922.0MHz	922.3MHz	922.6MHz	922.9MHz	923.2MHz	923.5MHz	923.8MHz	924.1MHz	924.4MHz	... ...	927.7MHz
Macro Channel Value :	Chn 65	Chn 66	Chn 67	Chn 68	Chn 69	Chn 70	Chn 71	Chn 72	Chn 73	Chn 74	Chn 75	... ...	Chn 86
config_words[2] bit :	bit 0	bit 1	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7	bit 8	bit 9	bit 10	... ...	bit 21

aaa-027301

Config words are used to enable/disable 192 kHz macro channels authorized for transmission.

Each macro channel is separated from another of 300 kHz. At least 9 macro channels must be enabled to ensure the minimum of 50 FCC channels (9 * 6 = 54). This function should be called each time you open the library, or your FCC configuration will not be applied.

Example: To enable Macro channel 1 to 9, that is to say 902.2 MHz to 904.8 MHz with 902.2 MHz as the main macro channel, you must set:

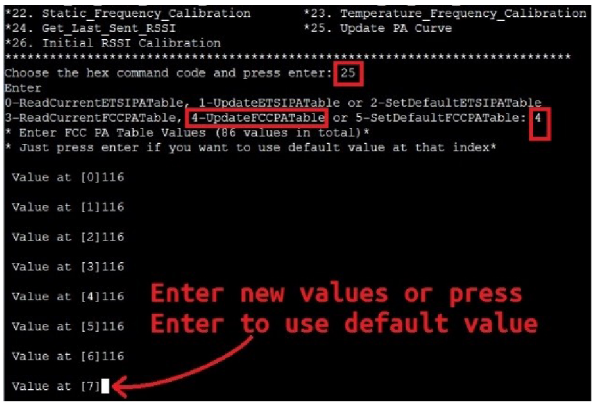
config_words[0] = [0x000001FF]
config_words[1] = [0x00000000]
config_words[2] = [0x00000000]

Figure 28. Sample frequency map for hopping in RCZ2/4

7.2.1.3 Update PA curve

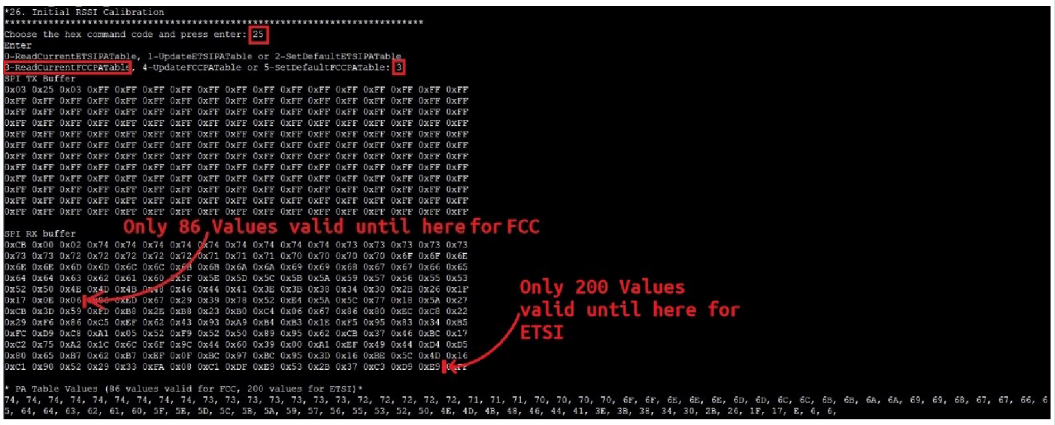
The current firmware supports the update of the output power amplitude curve for ramp-up and ramp-down functions:

1. Type **25** to select Update PA Curve command, and then press Enter.
2. Type **1** to update ETSI PA curve or type **4** to update FCC PA curve (for RCZ2/4), and then press Enter.
3. Type the new values and press Enter after each new value. *(To keep default value at the current index, just press Enter without entering a value. 86 values are needed for FCC table and 200 values are needed for ETSI table. After entering the last values and pressing Enter, the table will be updated with the new values, as shown in the following figure)*
4. To confirm the update of the PA curve, read the current updated PA curve by typing **25**, then press Enter. Type **0** (for ETSI table) or type **3** (for FCC table), then press Enter. *The first 3 bytes of the RX frame contain the frame length, error code and OL2385 state in order. The next 200 bytes contain the 200 updated PA curve values for ETSI. In case of FCC, only the next 86 values are valid out of the 200 values shown (as shown in the following figure)*



aaa-029132

Figure 29. Update PA curve values



aaa-029133

Figure 30. Read updated PA curve values

7.2.2 Legacy commands

The following is a set of essential commands are **no longer required** for OL2385 firmware versions after version number 01.11.01.20.03.20.17:

7.2.2.1 Temperature based frequency calibration

The next command for any device can be Temperature frequency calibration. This command also needs to be done only once after flashing new firmware on the OL2385. Temperature frequency calibration is optional. If you choose to skip this command, some TX and RX scenarios might not work. The crystal used on the boards has, sometimes, an initial offset in the generated frequency depending on its temperature. Therefore, the actual frequencies generated from the board differ slightly from the commanded frequency. For example, if a command to start an unmodulated continuous wave at 868.13 MHz is given, one can see on a spectrum analyzer that the peak of the spectrum is not exactly at 868.13 MHz, but is at an offset. This frequency offset can be compensated on the device by getting the temperature behavior curve from the crystal manufacturer. Such a curve over temperature with ppm values can be then sent to the OL2385 using the SPI command Temperature_frequency_calibration. These ppm values can be translated to absolute Hz value on OL2385. An example of such a curve for an

NDK crystal is shown in Table 9. A total of 26 signed integer values have to be sent in order to cover the temperature range of –40 to +85 °C.

Table 9. Crystal Temp vs. Frequency deviation curve

Calibration values	Temp [Reference temp. at +25°C]	Crystal 1	Crystal 2	Crystal 3	Average values	Rounded values
1	–40	–5.7	–4.4	–9.7	–6.6	–7
2	–35	–1.4	–0.2	–5.2	–2.3	–2
3	–30	2.1	3.2	–1.4	1.3	1
4	–25	4.7	5.8	1.5	4.0	4
5	–20	6.5	7.6	3.7	5.9	6
6	–15	7.6	8.7	5.2	7.2	7
7	–10	8.2	9.3	6.2	7.9	8
8	–5	8.1	9.2	6.4	7.9	8
9	0	7.6	8.6	6.2	7.4	7
10	5	6.7	7.4	5.5	6.5	7
11	10	5.4	6.0	4.5	5.3	5
12	15	3.7	4.2	3.2	3.7	4
13	20	1.9	2.1	1.7	1.9	2
14	25	0.0	0.0	0.0	0.0	0
15	30	–2.3	–2.4	–1.8	–2.1	–2
16	35	–4.3	–4.6	–3.5	–4.1	–4
17	40	–6.4	–6.9	–5.1	–6.1	–6
18	45	–8.2	–8.9	–6.6	–7.9	–8
19	50	–9.9	–10.7	–8.0	–9.5	–9
20	55	–11.1	–12.2	–8.9	–10.7	–11
21	60	–12.1	–13.3	–9.5	–11.6	–12
22	65	–12.3	–13.8	–9.6	–11.9	–12
23	70	–11.9	–13.3	–8.8	–11.3	–11
24	75	–11.1	–12.5	–7.6	–10.4	–10
25	80	–9.3	–10.9	–5.5	–8.6	–9
26	85	–6.7	–8.3	–2.5	–5.9	–6

Such a curve is usually provided by the crystal manufacturer for a set of a few crystals that differ in absolute values but show the same curve shape. One can build the mean value over all samples for every temperature (column) and round it to get integer values. These 26 rounded integer values have to be calculated only once for a specific crystal type. The same values can be used for every board as long as the crystal type is the same. For sending the values over SPI, start with the value at –40 °C and end with the value at +85 °C. Use the hex notation of these signed integer values. The SPI command number is 23 with a subcommand number as 1. If you would like to read the already stored values on OL2385, use the SPI command number 23 with 0 as the subcommand

number and look at the SPI RX buffer values starting from the 4th byte value until the 29th. With SPI command 23 and sub command as 2, you can reset the table inside the OL2385 to the default value of 0.

Note: *NXP currently supports boards operating a TCXO only and not a standard XTAL oscillator.*

Note:

- *This calibration has to be done once and done before doing the static frequency calibration.*
- *This calibration is useless for RCZ3, because TCXO is used as crystal.*
- *This calibration makes sense for final production or certification at SIGFOX. For normal RF evaluation, skip this and proceed to* Section 7.2.2.2 "Static frequency calibration"*, because temperature does not vary that much in normal test environment.*

7.2.2.2 Static frequency calibration

The next command for any device is Static frequency calibration. This command also needs to be done only once after flashing new firmware on the OL2385. The Static frequency calibration command is optional. If you choose to skip this command, some TX and RX scenarios might not work. The crystal used on the boards has, sometimes, an initial offset. Therefore, the actual frequencies from the board differ slightly from the commanded frequency. For example, if a command to start an unmodulated continuous wave at 868.13 MHz is given, on a spectrum analyzer the peak of the spectrum is not exactly at 868.13 MHz. It is at an offset as shown in Figure 31. This offset is variable for different crystals, but on a single board it is relatively fixed and is needed to be compensated. This compensation is not required for an RCZ3 device, because it uses a TCXO.

Note: *NXP currently only supports boards operating a TCXO and not a standard XTAL oscillator.*

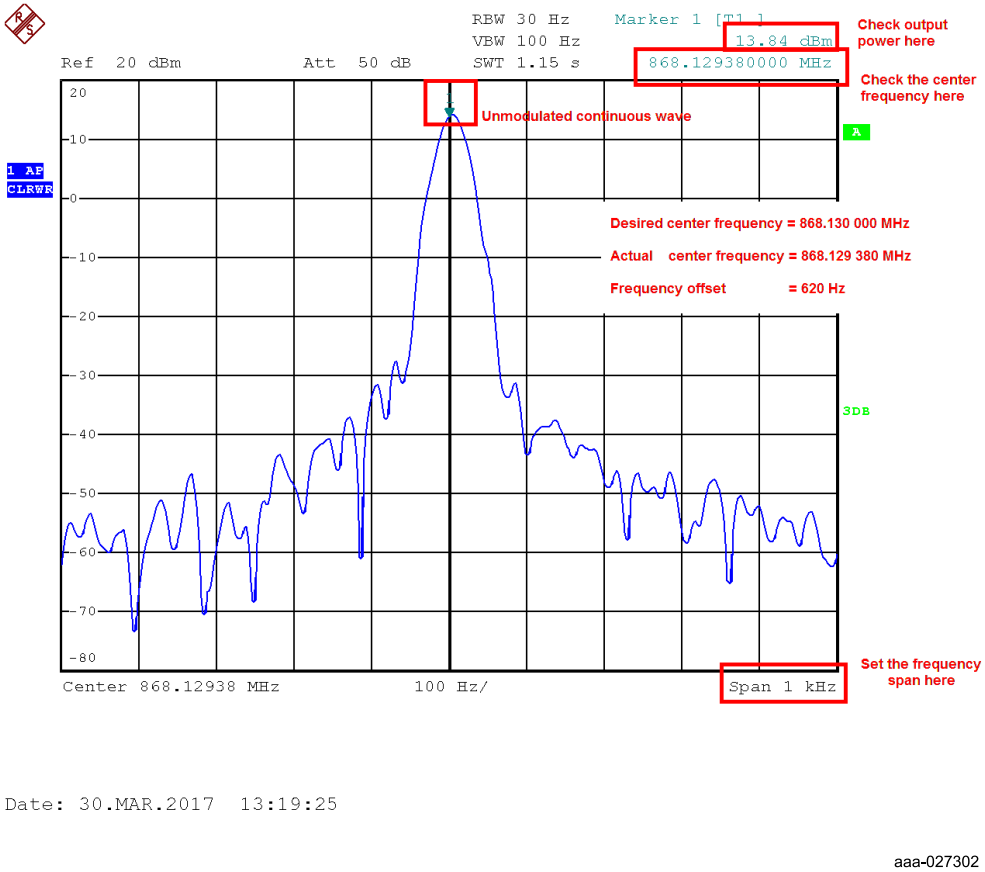


Figure 31. Offset calculation for static frequency calibration

The value of offset has to be calculated by using a continuous wave (SPI command number b) with a spectrum analyzer. Another method is to use 9 continuous frames on the same frequency (SPI command sequence 14-0-9) with a spectrum analyzer. There is also a way of getting this offset value using the SIGFOX SFXIQ tool. This value of offset is stored in nonvolatile memory of the OL2385. Therefore, the value does not need to be set again on reset. This command has to be executed again only if the device is flashed using a 2link box with new firmware.

Follow the following steps for static frequency calibration:

1. Press the reset button on the KL43Z to start from the beginning (optional).
2. Choose Send Wake up: Type **1**, and then press Enter. (This step is optional and not required if there is no device reset).
3. Choose the region Change to RCZXX: Type **15** or **16** or **17** or **18**. Press Enter. This will program the opening center frequencies of each region into the OL2385.
   Example of switching to RCZ1:
   Type **15**, and then press Enter. Type **0** for zero attenuation steps for uplink TX power, and then press Enter. Type 0 for 55.2 MHz crystal type, and then press Enter. Type **3** for SIGFOX repetition frames, and then press Enter. Type **0** for 14 dBm PA level, and then press Enter.
4. Choose the Static_frequency_calibration: Type **22**, and then press Enter. It will ask for adding or subtracting the offset. Type **2**, and then press Enter. It will ask for an offset value. Type **0**, and then press Enter. This will reset the offset first inside OL2385.
   Note: The offset always has to be an absolute value from the desired frequency.

You cannot fine tune it again and again by sending the values and expecting them to accumulate. The value is always the absolute difference from the desired frequency. If you see wrong results, it could be the entered value was wrong. That is why the value has to be reset.

5. Connect the RF connector of the OL2385 device to a spectrum analyzer using an SMA cable.
6. Choose the Continuous wave: Type **b**, and then press Enter. The CW should be ON. Write down the offset value from the spectrum analyzer as shown in the figure above.
7. Choose the Static_frequency_calibration: Type **22**, and then press Enter. It will ask for adding or subtracting the offset. Choose 1 or 2 according to the offset, and then press Enter. It will ask for an offset value. Type the offset value without the sign, and then press Enter. The sign was chosen by 1 or 2. Now the device is calibrated.
8. Choose the region Change to RCZXX: Type **15** or **16** or **17** or **18**. Press Enter. This will program the calibrated opening center frequencies of each region into the OL2385.
9. Choose the Continuous wave: Type **b**. Press Enter. The CW should be ON. Check if the peak is on correct center frequency.
   or
   After the compensation, you can also use the SPI command sequence of test mode 14-0-9 with max hold setting on trace of spectrum analyzer. This sequence will send nine frames on constant center frequency. Here you can also see the peak of the spectrum has now adjusted on correct frequency.

Note:

This method is used to compensate the crystal on the center frequency of a respective region. The device can be compensated on any other frequency using command Set_UL_frequency – 9 first to change the uplink frequency. Then, use the Continuous wave command to see the peak at this chosen frequency. The offset obtained from this peak will correspond to the chosen frequency. One can then use this offset for calibration in the above process. However, the proof of compensation cannot be checked by just starting the CW again. It is not the continuous wave command that actually takes the offset into calculation of frequency. The offset is taken into consideration by the SIGFOX library. Changing a zone uses those SIGFOX library functions and so does the test mode. Therefore, using Change to RCZXX or 14-0-9 will show the peak at compensated frequency value. But, these two work only with center frequencies of the region. The desired frequency is used to calculate the offset, then compensate it using the method above, but catching the proof of compensation on desired frequency is not possible. One can only see the proof on center frequencies of the region.

After running five commands, any command can be sent. For example, Send payload can be used to send three SIGFOX frames on three different frequencies. Continuous wave can be used for getting a constant unmodulated wave. After running above five commands, one can now choose to send any command as required. For example, Send payload can be used to send three SIGFOX frames on three different frequencies. Continuous wave can be used for getting a constant unmodulated wave.

7.2.3 Miscellaneous commands

The following sections contain commands that are not mandatory, but useful in certain conditions.

7.2.3.1 Choosing ciphering key

The next command is to choose the ciphering key for the transmission. The private ciphering key is stored in the protected area of the memory within the IC. The messages enciphered with this key will only be received and decoded properly by a real base station. In order to test it with the SNEK emulator tool, use the Switch to public key – 1a command next. In order to test with SFXIQ tool, make sure that the key used in the config.txt file is same as the private key, or else they also have to use Switch to public key as the next command. Perform this command once after the reset.

7.2.3.2 Set FCC macro channel

This command is only necessary for RCZ2 and RCZ4 devices because these regions use the frequency hopping algorithm on macro frequency channels. A frequency map is actually used to enable frequencies on which the device can hop for transmitting frames. The channels in this map are enabled using config word settings. See the SIGFOX_api.h file for detailed information.

Set the FCC Macro Channel

Example:

config_words[0] = [0x000001FF]

config_words[1] = [0x00000000]

config_words[2] = [0x00000000]

default_SIGFOX_channel = 1

Macro Channel Value MHz :	902.2MHz	902.5MHz	902.8MHz	903.1MHz	903.4MHz	903.7MHz	904.0MHz	904.3MHz	904.6MHz	904.9MHz	905.2MHz	... ...	911.5MHz
Macro Channel Value :	Chn 1	Chn 2	Chn 3	Chn 4	Chn 5	Chn 6	Chn 7	Chn 8	Chn 9	Chn 10	Chn 11	... ...	Chn 32
config_words[0] bit :	bit 0	bit 1	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7	bit 8	bit 9	bit 10	... ...	bit 31
Macro Channel Value MHz :	911.8MHz	912.1MHz	912.4MHz	912.7MHz	913.0MHz	913.3MHz	913.6MHz	913.9MHz	914.2MHz	914.5MHz	914.8MHz	... ...	921.1MHz
Macro Channel Value :	Chn 33	Chn 34	Chn 35	Chn 36	Chn 37	Chn 38	Chn 39	Chn 40	Chn 41	Chn 42	Chn 43	... ...	Chn 64
config_words[1] bit :	bit 0	bit 1	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7	bit 8	bit 9	bit 10	... ...	bit 31
Macro Channel Value MHz :	921.4MHz	921.7MHz	922.0MHz	922.3MHz	922.6MHz	922.9MHz	923.2MHz	923.5MHz	923.8MHz	924.1MHz	924.4MHz	... ...	927.7MHz
Macro Channel Value :	Chn 65	Chn 66	Chn 67	Chn 68	Chn 69	Chn 70	Chn 71	Chn 72	Chn 73	Chn 74	Chn 75	... ...	Chn 86
config_words[2] bit :	bit 0	bit 1	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7	bit 8	bit 9	bit 10	... ...	bit 21

aaa-029134

Figure 32. Macro channel table

7.2.3.3 Reset FCC Macro channel

This command is only necessary for RCZ2 and RCZ4 devices, because these regions use the frequency hopping algorithm as mentioned in Section 7.2.3.2 "Set FCC macro channel". One has to always take care to go back to default macro channels for their individual region, if there are no more free channels left for transmitting the frames. The free channel information can be obtained by using the Get_FCC_macro_channel SPI command. If the returned value is not equal to 0x30, then one might have to call the Reset_FCC_Macro_Channel to restore the default macro channel settings. Therefore, before sending any SIGFOX frame transmission command over SPI for RCZ2/4, the MCU must send a Get_FCC_macro_channel to check the free macro channels first and then call Reset_FCC_Macro_Channel, if required.

Notes:

- *This command does not have to be performed for RCZ1 or RCZ3.*

- *This command is not necessary for RF evaluation of the board, but can be the root cause of not seeing the transmission of the frame over the radio, even though there is no error reported.*

7.3 Frequently used RF test cases using SPI commands

Check the Section 5.4.1 "Setting up KL43Z board with terminal program" and Section 7.2 "OL2385 Transceiver Initialization Sequence" before you proceed to start performing the following tests.

7.3.1 Unmodulated continuous wave

1. Press the reset button on the KL43Z to start from the beginning (optional).
2. Choose Send Wake up: Type **1**, and then press Enter. (This step is optional and not required if there is no device reset).
3. Choose the region "Change to RCZXX": Type **15** or **16** or **17** or **18**, and then press Enter.
4. The next steps will program the opening center frequencies of each region into the OL2385.
   a. Type **0** for zero attenuation steps for uplink TX power, and then press Enter.
   b. Type **0** for 55.2 MHz crystal type or type **1** for TCXO, and then press Enter.
   c. Type **3** for SIGFOX repetition frames, and then press Enter.
   d. Type **0** for 14 dBm PA level, and then press Enter.
5. Choose Continuous wave command: Type **b**, and then press Enter. Now type **1**, and then press Enter to start continuous wave. The CW should be on and you can check this on a spectrum analyzer.
6. If you want to see the continuous wave on a different frequency, choose Continuous wave command: Type **b**, and then press Enter. Now type **0**, and then press Enter to stop the continuous wave first. The CW should be off and you can check this on a spectrum analyzer.
7. Use the command Set UL Frequency: Type **9** press Enter, type frequency value in Hz and press Enter. If you do not want to change the frequency, move to the next step.
8. Choose Continuous wave command: Type **b**, and then press Enter. Now type **1**, and then press Enter to start continuous wave. The CW should be on and you can check this on a spectrum analyzer.

Note: *If you want to see the CW on a different frequency, stop the CW first. Then change the frequency to your desired frequency and then start CW again.*

7.3.2 Modulated continuous wave

1. Press the reset button on the KL43Z to start from the beginning.
2. Choose Send Wake up: Type **1**, and then press Enter.
3. Choose the region "Change to RCZXX". Type **15** or **16** or **17** or **18**, and then press Enter.
4. The next steps will program the opening center frequencies of each region into the OL2385.
   a. Type **0** for zero attenuation steps for uplink TX power, and then press Enter.
   b. Type **0** for 55.2 MHz crystal type or type **1** for TCXO, and then press Enter.
   c. Type **3** for SIGFOX repetition frames, and then press Enter.

d. Type **0** for 14 dBm PA level, and then press Enter.

5. If you want to see the continuous modulated wave on a different frequency, use the command Set UL Frequency: Type **9**, and then press Enter, type frequency value in Hz and press Enter. If you do not want to change the frequency, move to the next step.
6. Choose PSK_ASK_Zero_Sequence or PSK_ASK_AAAA_Sequence command: Type **1b** or **1C**, and then press Enter. The modulated CW should be on and you can check this on a spectrum analyzer. There is no way to turn off the CW except the reset of the device.
7. If you want to change the frequency now, repeat the steps from the beginning of the sequence. There is no way to change the frequency on-the-fly.

7.3.3 Sending SIGFOX frames (frequency hopping)

1. Press the reset button on the KL43Z to start from the beginning (optional).
2. Choose Send Wake up: Type **1**, and then press Enter. (This step is optional and not required if there is no device reset)
3. Choose the region "Change to RCZXX": Type **15** or **16** or **17** or **18**, and then press Enter.
4. The next steps will program the opening center frequencies of each region into the OL2385.
   a. Type **0** for zero attenuation steps for uplink TX power, and then press Enter.
   b. Type **0** for 55.2 MHz crystal type or type **1** for TCXO, and then press Enter.
   c. Type **3** for SIGFOX repetition frames, and then press Enter.
   d. Type **0** for 14 dBm PA level, and then press Enter.
5. Choose Switch to Public Key, if not testing with real a base station. Type **1a**, and then press Enter.
6. Choose Static frequency calibration (Optional, see Section 7.2 "OL2385 Transceiver Initialization Sequence"): Type **22**, and then press Enter; Type **1** or **2**, and then press Enter; Type frequency offset value, and then press Enter.
7. Choose Send payload: Type **4**, and then press Enter; type a payload length up to 12 bytes, and then press Enter; type payload hex bytes and press Enter.

7.3.4 Sending SIGFOX frames (constant carrier)

1. Press the reset button on the KL43Z to start from the beginning (optional).
2. Choose Send Wake up: Type **1**, and then press Enter (This step is optional and not required if the device has not had a recent reset or it is already awake).
3. Choose the region "Change to RCZXX": Type **15** or **16** or **17** or **18**, and then press Enter.
4. The next steps will program the opening center frequencies of each region into the OL2385.
   a. Type **0** for zero attenuation steps for uplink TX power, and then press Enter.
   b. Type **0** for 55.2 MHz crystal type or type **1** for TCXO, and then press Enter.
   c. Type **3** for SIGFOX repetition frames, and then press Enter.
   d. Type **0** for 14 dBm PA level, and then press Enter.
5. Choose Switch to Public Key, if not testing with a real base station. Type **1a**, and then press Enter.

6. Choose Static frequency calibration (Optional, see the section Controlling OL2385 using terminal program): Type **22**, and then press Enter; Type **1** or **2**, and then press Enter; Type **frequency offset value**, and then press Enter.
7. Choose Send test mode: Type **14**, and then press Enter; type **0** as test mode, and then press Enter; type number of frames as test mode config and press Enter. Check the section Testing for P1 SIGFOX certification (SFXIQ tool) for details about test modes.

7.3.5 LBT testing for RCZ3

This functionality is only tested for ARIB standard in RCZ3 devices for the Listen Before Talk feature. It means that the device should listen to the 200 kHz SIGFOX band during a sliding window set before starting to send a SIGFOX frame on radio. If the channel is clear (below a certain threshold of signal level, typically –80 dbm) during the continuous minimum carrier sense time (cs_min = 5 ms), the frame can be sent. Otherwise, it continues to listen to the channel until the expiration of the carrier sense maximum window timer. If the timer has expired, the device can retry to send the frame until the maximum number of attempts has been reached. With that being said, it is important to note that each SIGFOX send payload command actually sends the same payload on three different frames on three different frequencies. These frames are called *frame 1*, *frame 2* and *frame 3*. The first frame is attempted first to be successfully sent. If the first frame itself fails the transmission after trying the maximum number of times, frame 2 and frame 3 are not sent. Otherwise, frame 2 and frame 3 are sent with the same carrier sensing procedure. The only difference is the carrier sensing maximum window timer runs for both frame 2 and 3 together instead of individually.

Figure 33. LBT algorithm for RCZ3

The number of retry attempts can be set by setting config words by the SPI command Set_FCC_Macro_Channel. The name Set_FCC_Macro_Channel is a bit of a misnomer, because the command is actually used for setting config words. Only in the case of RCZ2/4 is it used for setting macro channels for frequency hopping. In the case of RCZ3, the config words are used to set the LBT retry attempt number among other configurations. The config words have no meaning for RCZ1. The detailed information about config word can be reached by looking into SIGFOX_api.h header file inside the code (www.nxp.com/OM2385).

The following tools are needed in order to perform this test:

- Vector signal generator, which will generate the busy signal or scrambling on the RCZ3 TX frequency

- RF connector cable connected to the OL2385 and the vector signal generator

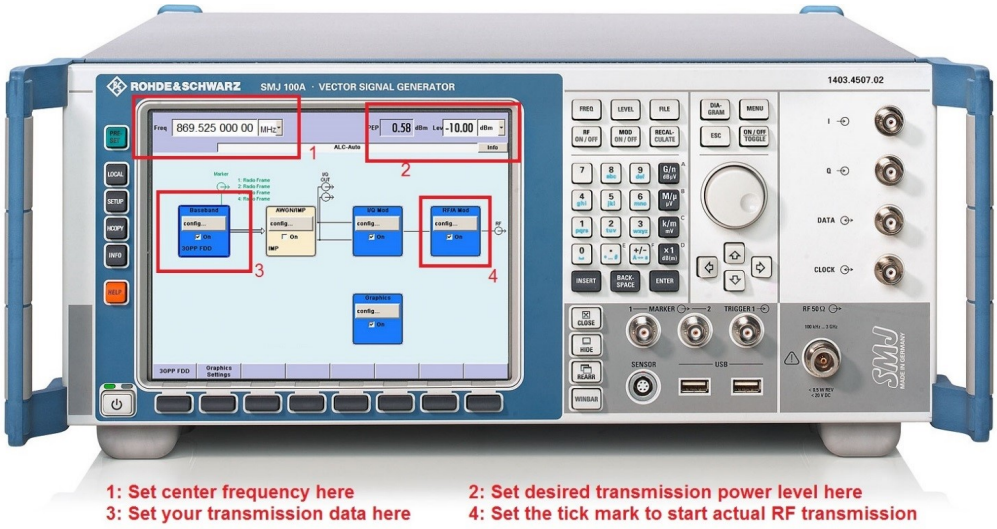
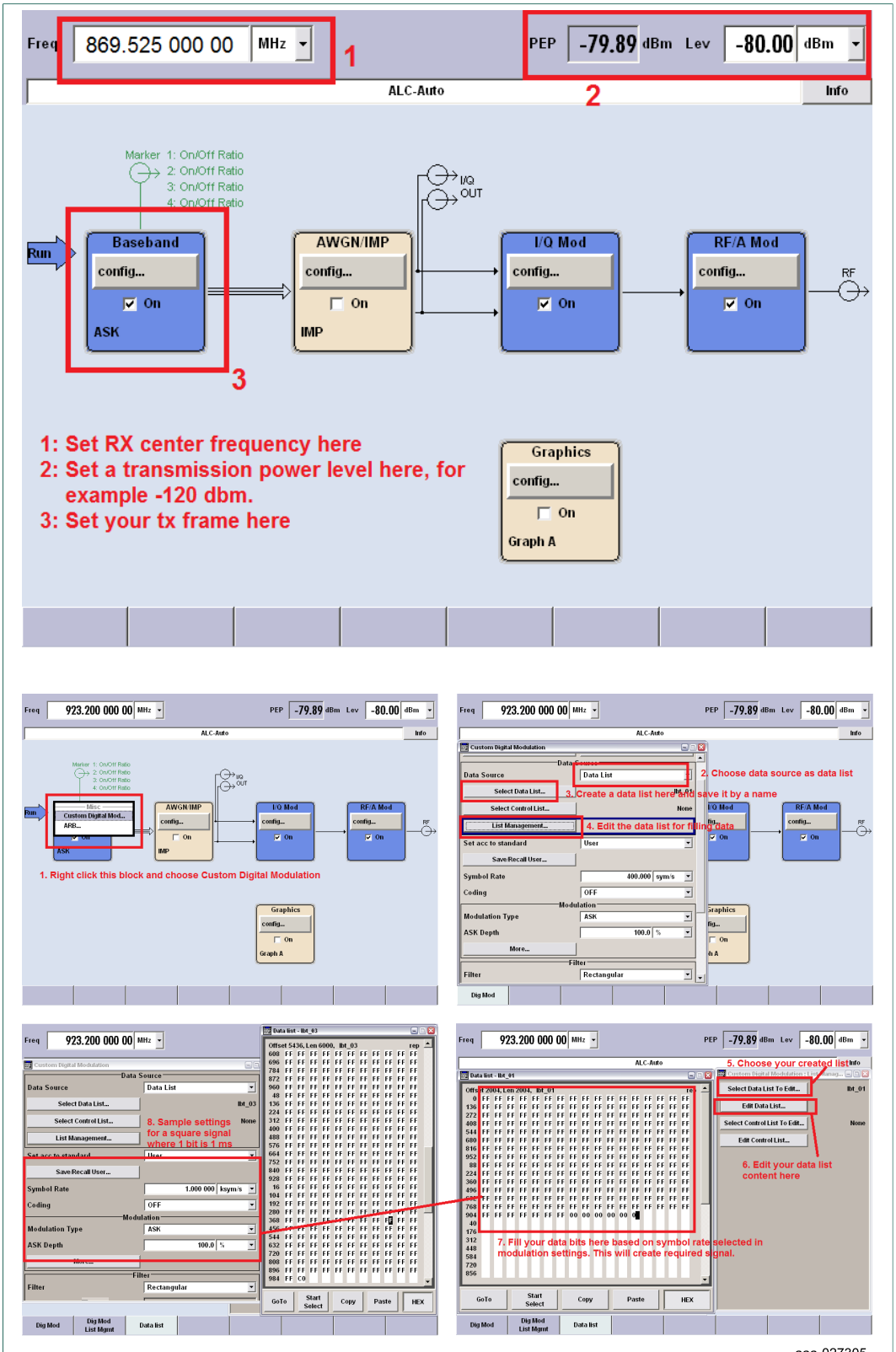


Figure 34. Signal generator used for LBT test

There are three test setups that are needed to be tested for a complete LBT testing. Each setup requires a special square test signal for creating a busy condition on the channel. Before we start with the description of the test setups, here is an example of a test signal creation using the vector signal generator shown in Figure 34. The following steps can be taken to generate a signal:

Step1: Press Preset on the vector signal analyzer.

Step 2: Follow the steps explained in the diagrams below in sequence.



aaa-027305

Figure 35. Signal generator settings

LBT Test Setup 1: All frames pass

Signal type: Square signal

Period: 5 s

Duty Cycle: 2 %

This means that the signal will be high (–80 dBm power level) during 4.9 s and low 0.1 s (total duration 5 s). When executing this test with the device, it should be able to transmit all three frames without any problem. This is proof that if a window of more than continuous 5 ms is found with less than –80 dBm power level, the channel is considered as free and transmission should take place. Because we have 0.1 s or 100 ms duration with low power signal, successful transmission should take place in all three frames as shown below.

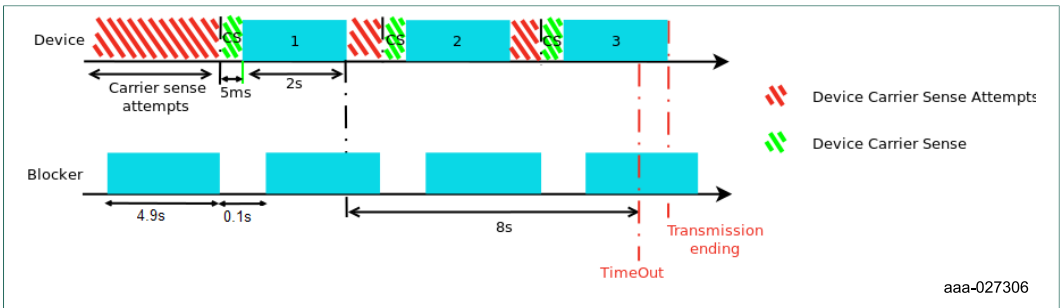


Figure 36. LBT test - All frames pass

LBT Test Setup 2: No frames pass

Signal type: Square signal

Period: 5 s

Duty Cycle: 0.1 %

This means that the signal will be high (–80 dBm power level) during 4.995 s and low 5 ms (total duration 5 s). When executing this test with the device, it should not be able to transmit the three frames. This is a proof that if a window of exactly continuous 5 ms or less is found with less than –80 dBm power level, the channel is still considered as NOT free and transmission should not take place. Because there is a 5 ms duration with a low power signal, successful transmission of all three frames should not take place.

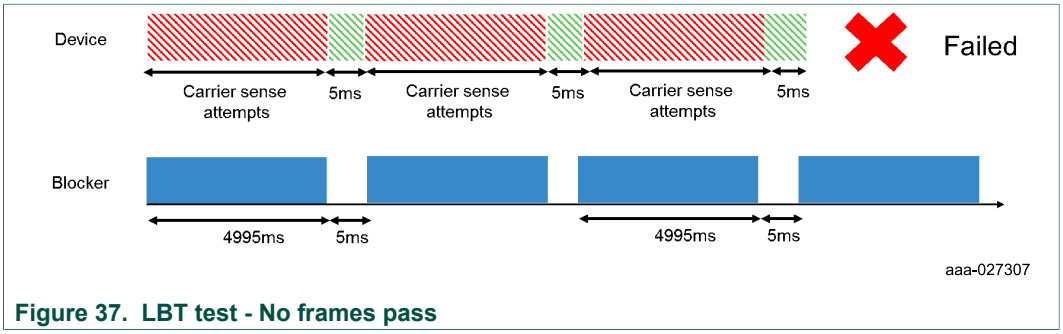


Figure 37. LBT test - No frames pass

Test Setup 3: Frames 1 and 2 pass, but frame 3 does not pass

Signal type: Square signal

Period: 10 s

Duty Cycle: 0.1 %

This means that the signal will be high (–80 dBm power level) during 9.990 s and low 10 ms (total duration 10 s). When executing this test with the device, it should be able to transmit the first frame and second frame only. According to the LBT concept, after frame 1 has been successfully transmitted, frame 2 and frame 3 have combined a total of only 8 s to find a free channel and get transmitted. In this test setup as frame 2 will find a free channel and gets transmitted, the duration of 8 s will be over by it getting transmitted and therefore frame 3 does not have time left to get transmitted. According to the timing diagram below, because we have no time left after frame 2 transmission, successful transmission of frame 3 should not take place.

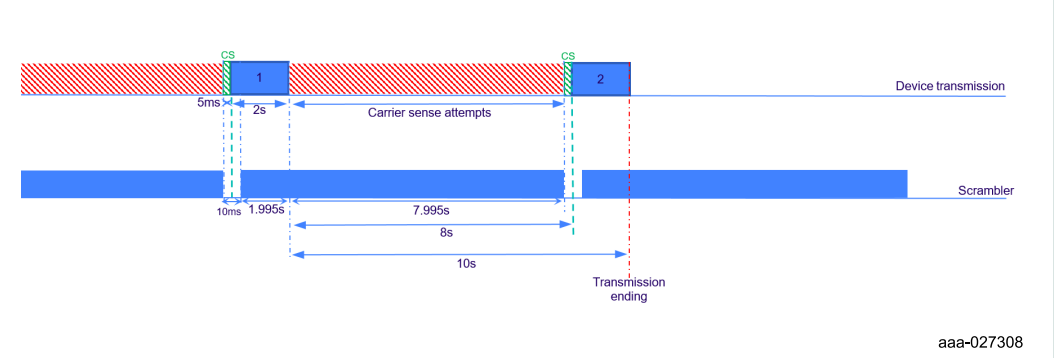


Figure 38. LBT test - Only frames 1, 2 pass

Steps for LBT testing:

1. Press the reset button on the KL43Z to start from the beginning. (optional)
2. Choose Send Wake up: Type **1**, and then press Enter. (This step is optional and not required if the device has not had a recent reset or it is already awake)
3. Take an RCZ3 board and choose the region Change to RCZ3: Type **17**, and then press Enter. This will program the opening center frequencies of RCZ3 region into the OL2385.
4. Prepare any one of your test signals as per the explanation in the previous sections (frequency, power level, test signal) on the vector signal generator. Connect the signal generator RF output to RF input of the OL2385 based device.
5. Turn on the signal by ticking on the check mark on the block RF-A mod on the signal generator.
6. Choose **Send payload**: Type **4**, and then press Enter; type a payload length up to 12 bytes, and then press Enter; type **payload hex bytes** and press Enter.
7. Check the SPI RX buffer bytes on the terminal. If test setup 1 or 3 was used, the second byte should be 0. If test setup 2 was used, second byte should be 0x3Bu. Detailed level of proof can only be checked using breakpoints inside the code to track transmitted frame numbers.

7.3.6 Receiver sensitivity testing (Test mode 3)

This test is also referred to as *RX sensitivity test*, which aims to check that your end product has no issue in the RX path and is able to demodulate at 600 bps GFSK. This is mentioned in the test document UNBT Test plan (PTP_UNBT_production.pdf located at nxp.com/OM2385). This test can be useful when one does not have SFXIQ SIGFOX certification tool to check the sensitivity. The test includes activating the OL2385 receiver at the center RX frequency of a region by invoking the test mode 3 via SPI command and then monitoring on an extra terminal program for the pass/fail criteria of frames and their measured RSSI level. The frames are also needed to be sent on the set RX center

frequency by a device (in our case we use a signal generator) using 600 bps GFSK modulation as per SIGFOX specification. In order to perform this test the following tools are also required:

- Extra FTDI cable to get the serial output at P20 of the OL2385 board. This output displays if the received frame is a pass or a fail and its measured RSSI value.
- A second terminal program, such as Putty, to connect to the above mentioned FTDI cable and to display the serial output.
- Vector signal generator with an RF connector cable connected to our device, which will generate and transmit the downlink test frame, the device will receive.

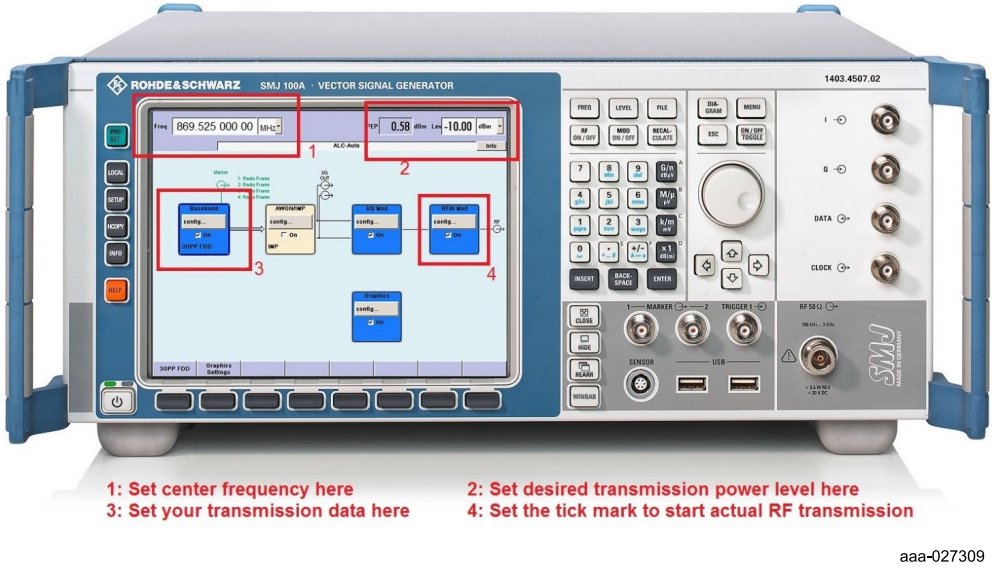


Figure 39. Signal generator used for receiver test

Once the additional tools above are prepared, the following steps have to be performed to start the test:

1. Press the reset button on the KL43Z to start from the beginning. (optional)
2. Choose Send Wake up: Type **1**, and then press Enter. (This step is optional and not required if the device has not had a recent reset or it is already awake)
3. Choose the region Change to RCZXX: Type **15** or **16** or **17** or **18**, and then press Enter. This will program the opening center frequencies of each region into the OL2385.
4. Choose Switch to Public Key, if not testing with a real base station: Type **1a**, and then press Enter.
5. Choose Static frequency calibration (optional, see Section 7.2 "OL2385 Transceiver Initialization Sequence"):
   a. Type **22**, and then press Enter
   b. Type **1** or **2**, and then press Enter
   c. Type **frequency offset value**, and then press Enter.
6. The receiver will be turned on by activating the test mode 3 in the next steps. In order to test, send a GFSK frame at a correct frequency. Check the RX center frequency of your region (see Section 7.3.2 "Modulated continuous wave" for region frequencies) and enter the value in the Freq field of the signal generator.
7. Fill the required power level on which you would like to transmit your test frames on the Lev field of the signal generator. This power level will be displayed on the P20

serial output of the OL2385 board, as the RSSI level of each received frame. In an ideal lossless system, displayed RSSI level value will be equal to the set power level value. However, that is not the case due to losses in the connected RF cable. A reduced value of the RSSI level on putty terminal is likely.

8. Click on the **config...** tab on the Baseband block of the block diagram on the signal generator. Choose the Custom digital modulation option from the list. This is to create the 2GFSK settings and required frame data, which is supposed to be transmitted.
9. Set the modulation settings as shown in Figure 41. These settings should be exactly the same as given for RX demodulation in SIGFOX specification PRS_UNB_MODEM_RCZX.pdf (location: nxp.com/OM2385).
10. Select data list as the data source and fill the GFSK frame AA AA B2 27 1F 20 41 84 32 68 C5 BA 53 AE 79 E7 F6 DD 9B in the data list as shown in the figure below. (Check the figures in Section 7.3.5 "LBT testing for RCZ3" for RCZ3 to know more about data list creation)
11. The settings for sending the frame on the right frequency are done now. Connect an RF cable from the signal generator to the OL2385.
12. Connect the P20 (marked on the backside of the board or within schematics) on the OL2385 board with RXD of the FTDI cable as shown in the figure below (usually a yellow wire). Connect the other end of the USB cable to the PC.
13. Download a terminal program of your choice. This example is the terminal program Putty.
14. Open the terminal program.
15. Fill the putty settings as per the diagram below (com port number and speed are main settings). Click on **Open** to start the putty program.
16. Choose **Send test mode SPI command**: Type **14**, and then press Enter; type **3** as test mode, and then press Enter; type number of frames as test mode config... and press Enter. This will start the receiver of the device, which waits for the frames to be received.
17. The device is now listening and waiting for the frames. Click on the check box on the RF/A Mod block of the block diagram on the signal generator to start RF transmission of the frame.
18. If all of the above steps were completed correctly, Passed RSSI = –80 dBm or Failed RSSI = –80 dBm will appear on the putty terminal as shown in Figure 43. This means the device successfully received the frames. If you do not see anything, the device is not correctly calibrated for static frequency, is not matched correctly for RX, or the transmission power level is too low.

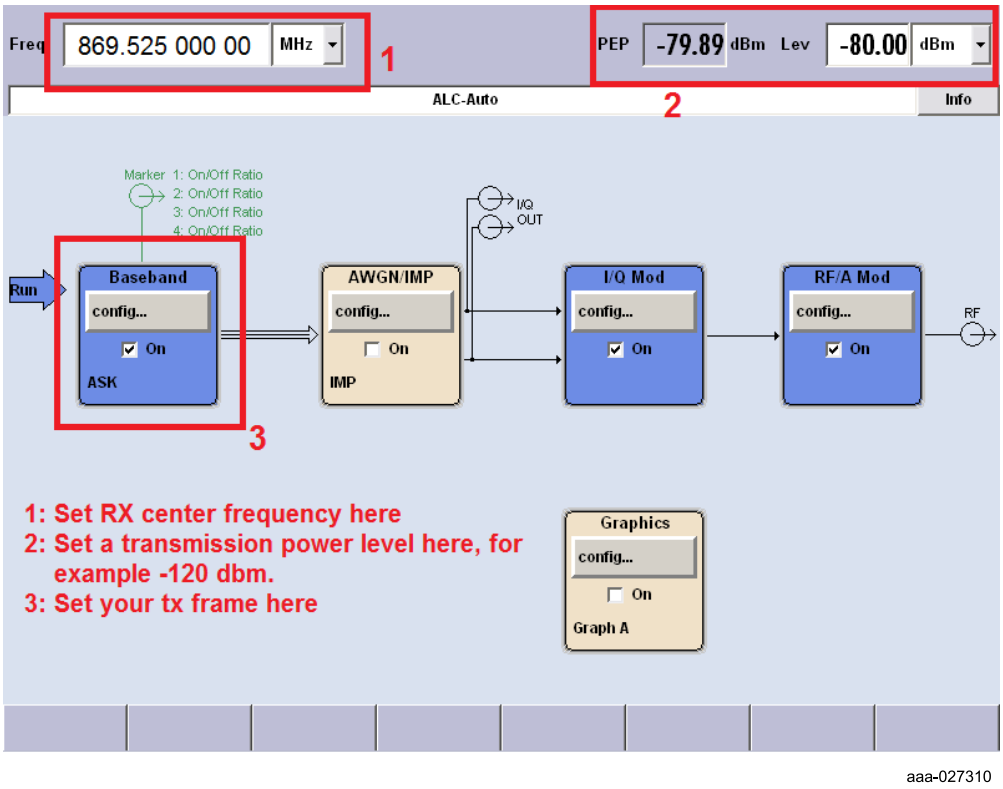


Figure 40. Signal generator settings for RX sensitivity test

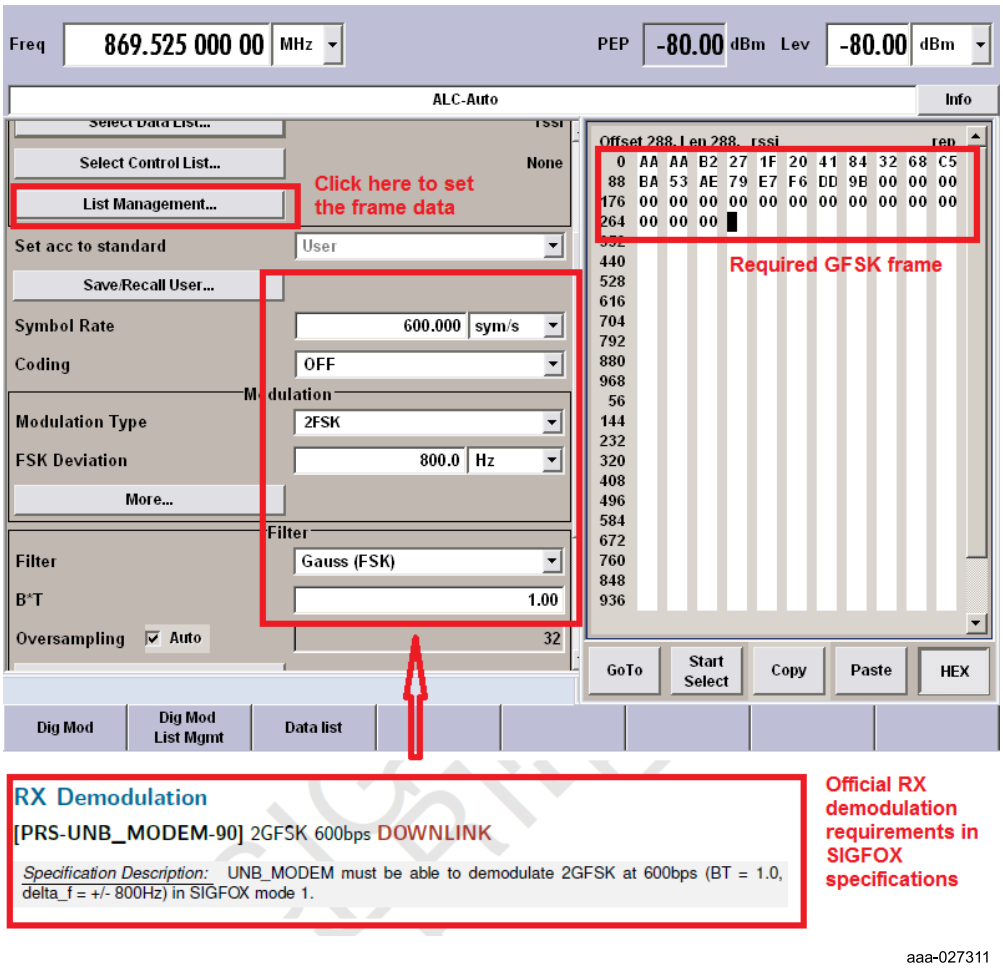


Figure 41. RX demodulation settings for sensitivity test

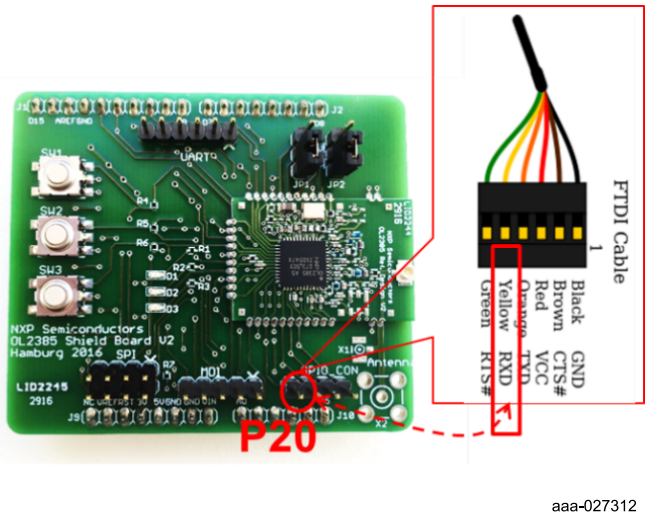


Figure 42. FTDI cable connection

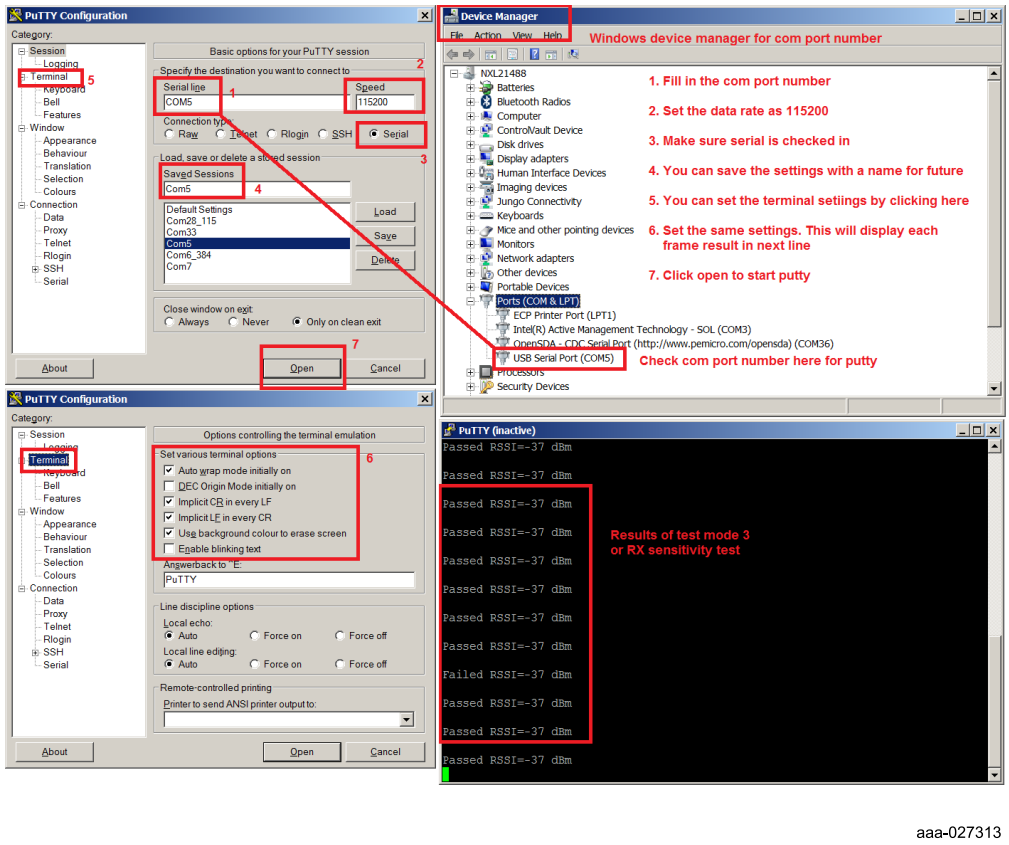


Figure 43. Putty settings for RX sensitivity test

7.3.7 How to get ID-PAC of a device

1. Press the reset button on the KL43Z to start from the beginning. (This step is optional and not required if the device has not had a recent reset or it is already awake)
2. Choose Send Wake up: Type **1**, and then press Enter.
3. Choose the region Change to RCZXX: Type **15** or **16** or **17** or **18**, and then press Enter. This will program the opening center frequencies of each region into the OL2385.
4. Choose Get info command: Type **8**, and then press Enter. The ID and PAC are displayed on the terminal.

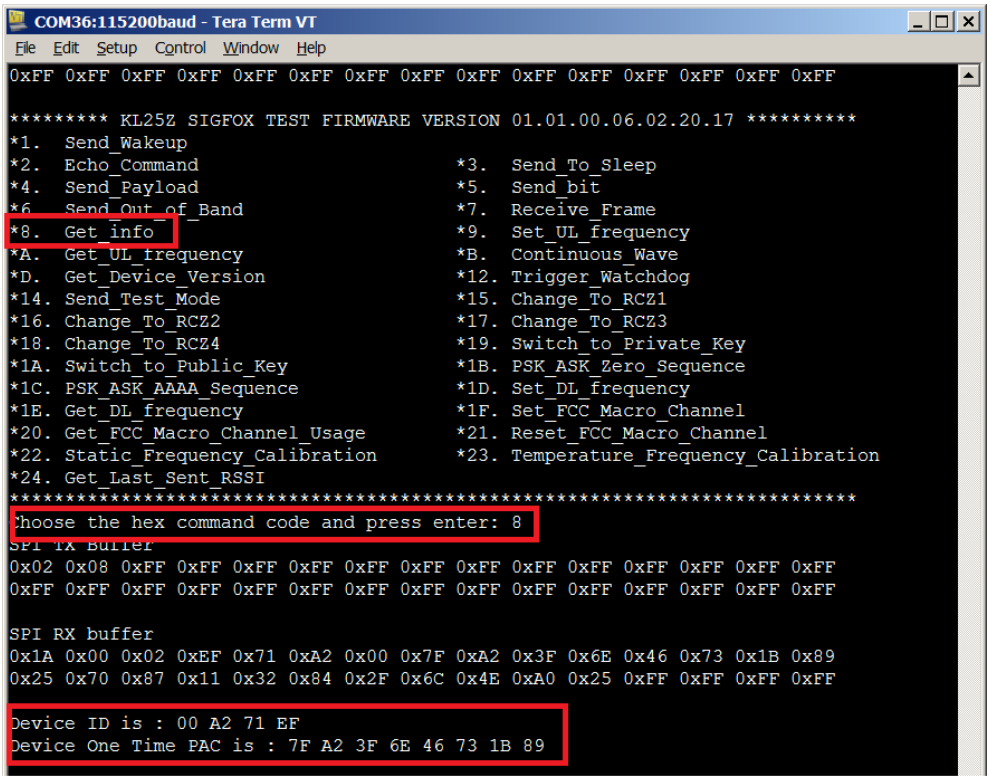


Figure 44. How to get ID-PAC from a device

7.4 Testing with SIGFOX network emulator kit

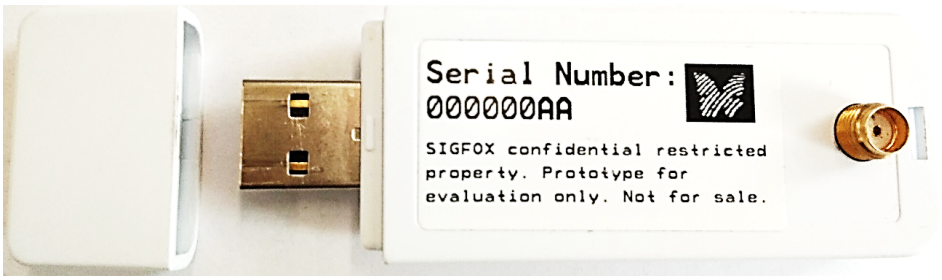
In order to facilitate the development of IoT applications, from the connected object to the application that processes the data transmitted by the objects, SIGFOX has developed a network emulator. This SNEK emulator (SIGFOX Network Emulator Kit) is a USB device associated to a software package that emulates the SIGFOX network. See Figure 45. It is intended solely for product or software designers for use in a research and development setting.

The emulator runs in conducted mode with the object under development and is compatible with all European and FCC bands, allowing the development of applications without concern for network coverage issues. The software package is downloadable. Potential protocol evolutions or new features can be taken into account by downloading the latest version of the software package.

1. Download the user guide and driver software for SNEK tool from: www.SIGFOX.com/en/support/download.
2. Install the executable file of the driver.
3. Connect the USB dongle to the PC and RF connector to your device.
4. Locate the folder **C:\Program Files (x86)\Snek** on your windows PC.
5. Start the application by running the snek.vbs file in a browser.
6. Follow the rest of the instructions from the latest SNEK user guide, which is available at the link provided in step 1. The user guide is updated from time-to-time by SIGFOX. It is recommended to follow instructions from the latest user guide and latest driver software.

The rest of the instructions include:

- Registering your device ID
- Choosing your region
- Setting up callback functions on backend emulator
- Receiving and viewing the successful SIGFOX message on aweb browser



aaa-027324

Figure 45. SNEK tool

8 Legal information

8.1 Definitions

Draft — The document is a draft version only. The content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included herein and shall have no liability for the consequences of use of such information.

8.2 Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors. In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory. Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification. Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products. NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Non-automotive qualified products — Unless this data sheet expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications. In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

Evaluation products — This product is provided on an "as is" and "with all faults" basis for evaluation purposes only. NXP Semiconductors, its affiliates and their suppliers expressly disclaim all warranties, whether express, implied or statutory, including but not limited to the implied warranties of non-infringement, merchantability and fitness for a particular purpose. The entire risk as to the quality, or arising out of the use or performance, of this product remains with customer. In no event shall NXP Semiconductors, its affiliates or their suppliers be liable to customer for any special, indirect, consequential, punitive or incidental damages (including without limitation damages for loss of business, business interruption, loss of use, loss of data or information, and the like) arising out the use of or inability to use the product, whether or not based on tort (including negligence), strict liability, breach of contract, breach of warranty or any other theory, even if advised of the possibility of such damages. Notwithstanding any damages that customer might incur for any reason whatsoever (including without limitation, all damages referenced above and all direct or general damages), the entire liability of NXP Semiconductors, its affiliates and their suppliers and customer's exclusive remedy for all of the foregoing shall be limited to actual damages incurred by customer based on reasonable reliance up to the greater of the amount actually paid by customer for the product or five dollars (US$5.00). The foregoing limitations, exclusions and disclaimers shall apply to the maximum extent permitted by applicable law, even if any remedy fails of its essential purpose.

Translations — A non-English (translated) version of a document is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

8.3 Trademarks

Notice: All referenced brands, product names, service names and trademarks are the property of their respective owners.

Kinetis — is a trademark of NXP B.V.

Tables

Figures

Contents

For more information, please visit: http://www.nxp.com
For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 6 April 2018