Getting started with the S2-LP development kits

Introduction

This document describes the development kits for the S2-LP device and related hardware and software components.

The S2-LP is an ultra low-power, low data rate, sub-1 GHz transceiver.

The following S2-LP development platforms are available:

1. STEVAL-FKI433V2 for 413-479 MHz
2. STEVAL-FKI868V2 and X-NUCLEO-S2868A2 for 826-958 MHz
3. STEVAL-FKI512V1 for 452-527 MHz
4. STEVAL-FKI915V1 for 902-928 MHz with external FEM
5. STEVAL-FKI001V1 dual radio BLE and sub-1 GHz development kit with BlueNRG-1 and S2-LP
6. X-NUCLEO-S2915A1 for the 915 MHz ISM frequency band
7. STDES-MONARCH reference design with BlueNRG-2 and S2-LP

1 Overview

This section describes all the software and hardware components of the S2-LP kits.

1.1 System requirements

The STSW-S2LP-DK application GUI has the following minimum requirements:

- PC with Intel® or AMD® processor running Windows (7, 8 or 10)
- At least 1 GB of free RAM
- USB ports
- 200 MB of available hard disk space
- Adobe Acrobat Reader 6.0 or above

1.2 STSW-S2LP-DK development kit setup

Launch the S2-LP DK-Setup-X.X.X.exe file and follow the onscreen instructions.

Note: IAR Embedded Workbench 8.32.1 or MDK ARM Keil V5.26.2.0 are required.

2 Hardware description

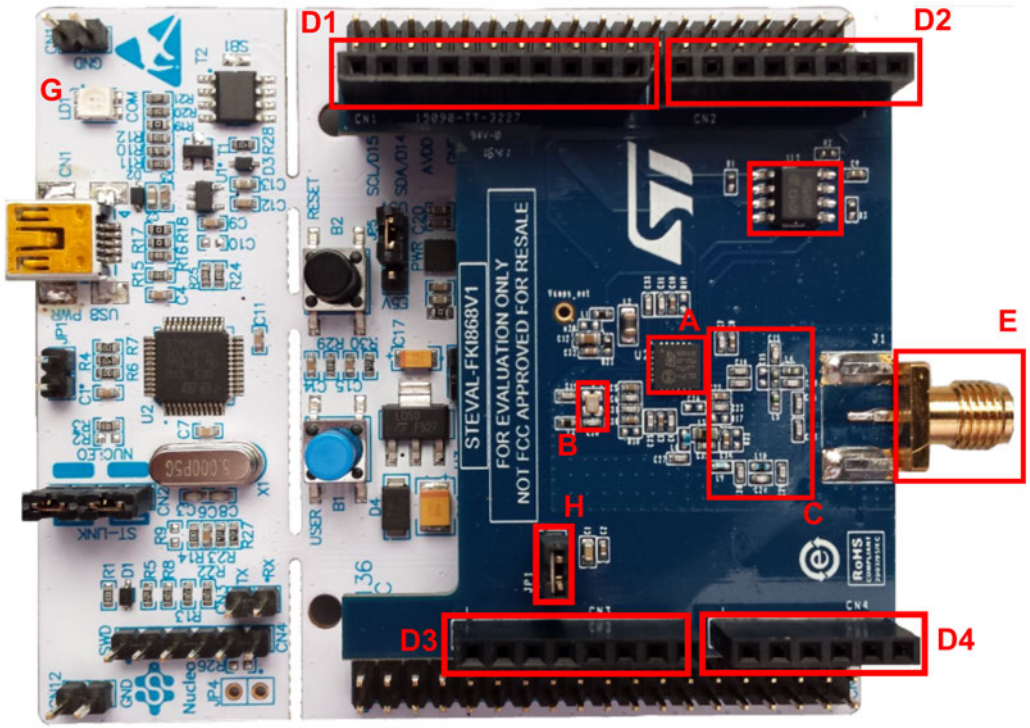
2.1 STEVAL-FKI868V2, STEVAL-FKI433V2 and STEVAL-FKI512V1 evaluation boards

The STEVAL-FKI868V2, the STEVAL-FKI433V2 and STEVAL-FKI512V1 evaluation boards are designed to work in the 826-958 MHz, in the 413-479 MHz and in the 452-527 MHz band, respectively.

Some features on the boards are (see Figure 1. STEVAL-FKI868V2 evaluation board features):

- S2-LP (A)
- 8 MHz high frequency crystal (B)
- Balun, matching network and harmonic filter (C)
- Two rows with Arduino compliant connectors (D1-4)
- SMA connector (E)
- An EEPROM to store the manufacturing data (F)
- A NUCLEO-L152RE or NUCLEO-L053R8 evaluation board (G)
- A jumper for S2-LP current measurement (H)

Figure 1. STEVAL-FKI868V2 evaluation board features



Pressing the reset button, the STM32 Nucleo development board resets.

2.1.1 S2-LP connections

S2-LP signal test points are split across two rows which are Arduino compliant connectors: CN1, CN3 and CN2, CN4.

The S2-LP shield is connected to the Nucleo motherboard via the Arduino compliant connectors.

The connectors and pin names below are used in the STEVAL-FKIxxxVx schematic diagram.

Table 1. S2-LP: FKI868V2 and FKI433V2 evaluation board pin description

Pin name	Pin number	Arduino connectors			
		CN1 (D1)	CN2 (D2)	CN3 (D3)	CN4 (D4)
VDD SMPS	1			pin 4	
SMPS1	2				
SMPS2	3				
XOUT	4				
XIN	5				
SDN	6		pin 8(SDN)		
VDDANASYNTH	7			pin 4	
VRSYNTH	8				
VREFVCO	9				
VDDVCOTX	10			pin 4	
TX	11				
VRRF	12				
RXN	13				
RXP	14				
VDDRXDIG	15			pin 4	
SDO	16	pin 5 (MISO)			
SDI	17	pin 4 (MOSI)			
SCLK	18		pin 4(SCK)		
CSN	19				pin 2 (CS)
GPIO0	20				pin 1 (GPIO0)
GPIO1	21				pin 3 (GPIO1)
GPIO2	22				pin 4 (GPIO2)
GPIO3	23				pin 6 (GPIO3)
VSMPS3	24				
GND	25	pin 7		pins 6 and 7	

The S2-LP evaluation board contains a discrete passive circuit for RF matching and balun and other additional components required by the S2-LP for proper operation.

2.1.2 STEVAL-FKI868V2, STEVAL-FKI433V2 and STEVAL-FKI512V1 power

S2-LPThe board can be powered by the Nucleo evaluation board mini USB connector.

When the JP1 jumper is fitted (H in Figure 1. STEVAL-FKI868V2 evaluation board features), the radio section is supplied.

By removing this jumper and connecting a power meter, you can measure the S2-LP current consumption.

2.2 STEVAL-FKI915V1 evaluation board

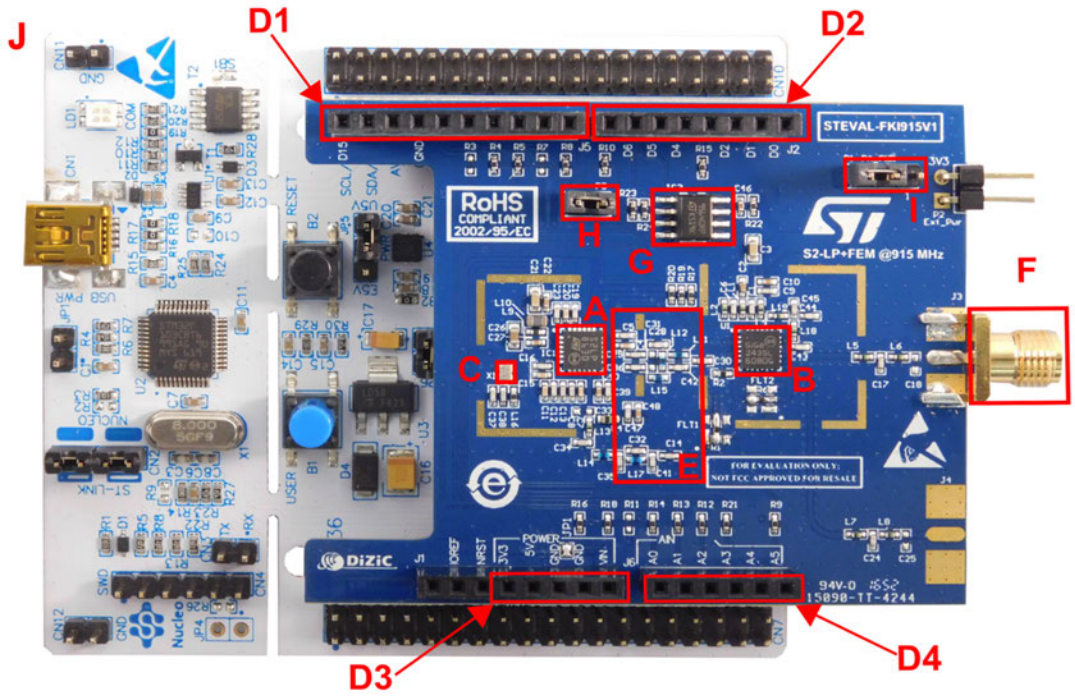
The STEVAL-FKI915V1 evaluation board is tuned to work for 904-1055 MHz frequency bands.

The STEVAL-FKI915V1 evaluation board features (see Figure 2. STEVAL-FKI915V1 evaluation board features):

- S2-LP (A)
- Skyworks SE2435L FEM (B)

- High frequency 8 MHz crystal (C)
- Two rows with Arduino compliant connectors (D1-D4)
- Balun, matching network and harmonic filter (E)
- SMA connector (F)
- An EEPROM to store the manufacturing data (G)
- A jumper for S2-LP current measurement (H)
- A jumper for Skyworks SE2435L FEM current measurement (I)
- A NUCLEO-L152RE or NUCLEO-L053R8 board (J)

Figure 2. STEVAL-FKI915V1 evaluation board features



2.2.1 S2-LP connections

S2-LP signal test points are split across two rows which are Arduino compliant connectors: CN1, CN3 and CN2, CN4.

The S2-LP shield is connected to the STM32 Nucleo motherboard using the Arduino compliant connectors.

The connectors and pin names below are used in the STEVAL-FKI915V1 schematic diagram.

Table 2. S2-LP: FKI915Vx evaluation board pin description

Pin name	Pin number	Arduino connectors			
		J5 (D1)	J2 (D2)	J1 (D3)	J6 (D4)
VDD SMPS	1			pin 4	
SMPS1	2				
SMPS2	3				
XOUT	4				
XIN	5				
SDN	6		pin 8 (SDN)		
VDDANASYNTH	7			pin 4	

Pin name	Pin number	Arduino connectors			
		J5 (D1)	J2 (D2)	J1 (D3)	J6 (D4)
VRSYNTH	8				
VREFVCO	9				
VDDVCOTX	10			pin 4	
TX	11				
VRRF	12				
RXN	13				
RXP	14				
VDDRXDIG	15			pin 4	
SDO	16	pin 5 (MISO)			
SDI	17	pin 4 (MOSI)			
SCLK	18		pin 4 (SCK)		
CSN	19				pin 2 (CS)
GPIO0	20				pin 1 (GPIO0)
GPIO1	21				pin 3 (GPIO1)
GPIO2	22				pin 4 (GPIO2)
GPIO3	23				pin 6 (GPIO3)
VSMPS3	24				
GND	25	pin 7		pins 6 and 7	

2.2.2 STEVAL-FKI915V1 power

The radio frontend is supplied via two different jumpers:

- P3 that supplies the S2-LP (H in Figure 2. STEVAL-FKI915V1 evaluation board features)
- P1 that supplies the Skyworks FEM (I in Figure 2. STEVAL-FKI915V1 evaluation board features)

The P1 jumper can be fitted in the following ways:

Table 3. STEVAL-FKI915V1 expansion board jumper description

P1 position	Comment
2-3	The FEM power supply is connected to the 3.3 V provided by the Nucleo motherboard
1-2	The FEM is supplied by an external voltage that can be provided by P2 connector pin 2.

To measure the radio part power consumption, add the two currents across P1 and P3.

2.3 X-NUCLEO-S2868A1

For X-NUCLEO-S2868A1 power up and S2-LP connections, refer to UM2405, "Getting started with the X-NUCLEO-S2868A1 Sub-1 GHz 868 MHz RF expansion board based on S2-LP radio for STM32 Nucleo", freely available at www.st.com.

2.4 X-NUCLEO-S2915A1

For X-NUCLEO-S2915A1 power up and S2-LP connections, refer to UM2641, "Getting started with the X-NUCLEO-S2915A1 Sub-1 GHz 915 MHz RF expansion board based on S2-LP radio for STM32 Nucleo", freely available at www.st.com.

2.5 STM32 Nucleo board

2.5.1 Push buttons

The board has one button to reset the microcontroller and another one available for the application.

2.5.2 LEDs

The available LEDs are:

- LD1: green (STM32 Nucleo embedded ST-LINK power on)
- LD2: green (user LED)
- LD3: red (microcontroller power)

2.5.3 Embedded ST-LINK

The ST-LINK/V2-1 programming and debugging tool is integrated in the STM32 Nucleo boards.

The ST-LINK/V2-1 makes the STM32 Nucleo boards "mbed" enabled.

2.5.4 STM32L053R8/STM32L152RE microcontroller

The STM32 Nucleo on-board microcontroller is programmed by the S2-LP DK firmware and is used to drive the device through the GUI or through the library examples.

2.6 STEVAL-IDB007V2 and STEVAL-IDB008V2 evaluation boards

The STEVAL-IDB007V2 and STEVAL-IDB008V2 evaluation boards key features are:

- three push buttons: one to reset the microcontroller and other two available for the application
- three LEDs:
  - LD1: green (STM32 Nucleo embedded ST-LINK power on)
  - LD2: green (user LED)
  - LD3: blue (microcontroller power)
- excellent receiver sensitivity: -88 dBm
- very low power consumption: 7.7 mA RX and 8.2 mA TX at +0 dBm
- new integrated balun: BALF-NRG-02D3 with matching network and harmonics filter
- embedded sensors:
  - 3D digital accelerometer
  - 3D digital gyroscope
  - MEMS pressure sensor with embedded temperature sensor
- battery holder
- on-board BlueNRG-1/BlueNRG-2 programmed by the S2-LP DK firmware and used to drive the device via GUI or library samples

2.7 STEVAL-FKI001V1 evaluation board

The STEVAL-FKI001V1 evaluation board key features are:

- two push buttons: one to reset the microcontroller and another one available for the application
- three LEDs:
  - LD1: green (STM32 Nucleo embedded ST-LINK power on)
  - LD2: green (user LED)
  - LD3: blue (microcontroller power)
- excellent receiver sensitivity: -88 dBm
- ultra-low power consumption: 7 mA RX and 10 mA TX at +10 dBm
- 50 Ω integrated balun: BALF-NRG-01D3 with matching network and harmonics filter, companion device of BlueNRG-1
- on-board BlueNRG-1 programmed by the S2-LP DK firmware and used to drive the device via GUI or library samples

2.8 Hardware setup

Step 1. Connect an antenna to the SMA connector

Step 2. Ensure the jumper configuration on the daughterboard is correct (see Section 2.1.2 STEVAL-FKI868V2, STEVAL-FKI433V2 and STEVAL-FKI512V1 power and Section 2.2.2 STEVAL-FKI915V1 power)

Step 3. Connect the motherboard to the PC through a USB cable

3 GUI software description

The S2-LP DK GUI included in the software package is a graphical user interface that can be used to interact with and evaluate the capabilities of the S2-LP device.

You can run this utility by clicking on the S2-LP GUI icon on the desktop or under: Start →STMicroelectronics →S2-LP DK X.X.X →S2-LP DK

This version of the GUI for S2-LP exclusively targets RF evaluation performance and only provides the RF test window and the manipulation of the device configuration parameters.

3.1 Installation

To use the S2-LP GUI, make sure you have correctly set up your hardware and software (S2-LP DK installed).

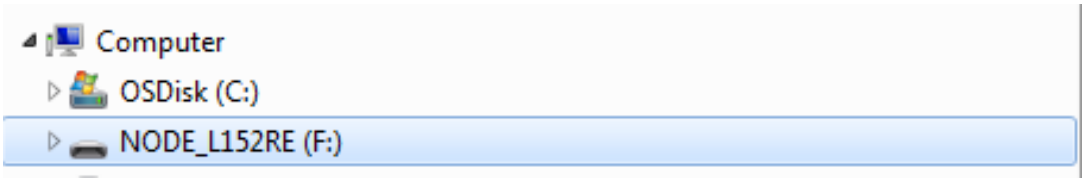
The firmware image to be programmed is available in the S2-LP DK SW package in the Binaries folder.

In order to download the firmware binary image into the internal Flash of the motherboard, follow these steps:

- Connect the motherboard to a PC USB port
- Open the S2-LP DK GUI
- Select the COM port associated to the board
- Open Tools → Firmware Upgrade and Browse to select the appropriate firmware, then press Open and wait for the firmware download.

Alternatively, it is possible to flash the motherboard by dragging and dropping the appropriate firmware into the disk drive recognized by Windows (for example, the drive F below).

Figure 3. NODE_L152RE disk drive



3.2 Detailed description

The S2-LP DK GUI can use only one S2-LP DK - MB plus S2-LP RF - DB connected through a USB cable to a PC. So, it is necessary to run one instance of S2-LP DK - GUI for each board connected to the PC.
Figure 4. Connection setup 1: 1 PC with S2-LP-DK GUI and Figure 5. Connection setup 2: 2 PCs with S2-LP-DK GUI show typical connections with one or two PCs.

Figure 4. Connection setup 1: 1 PC with S2-LP-DK GUI

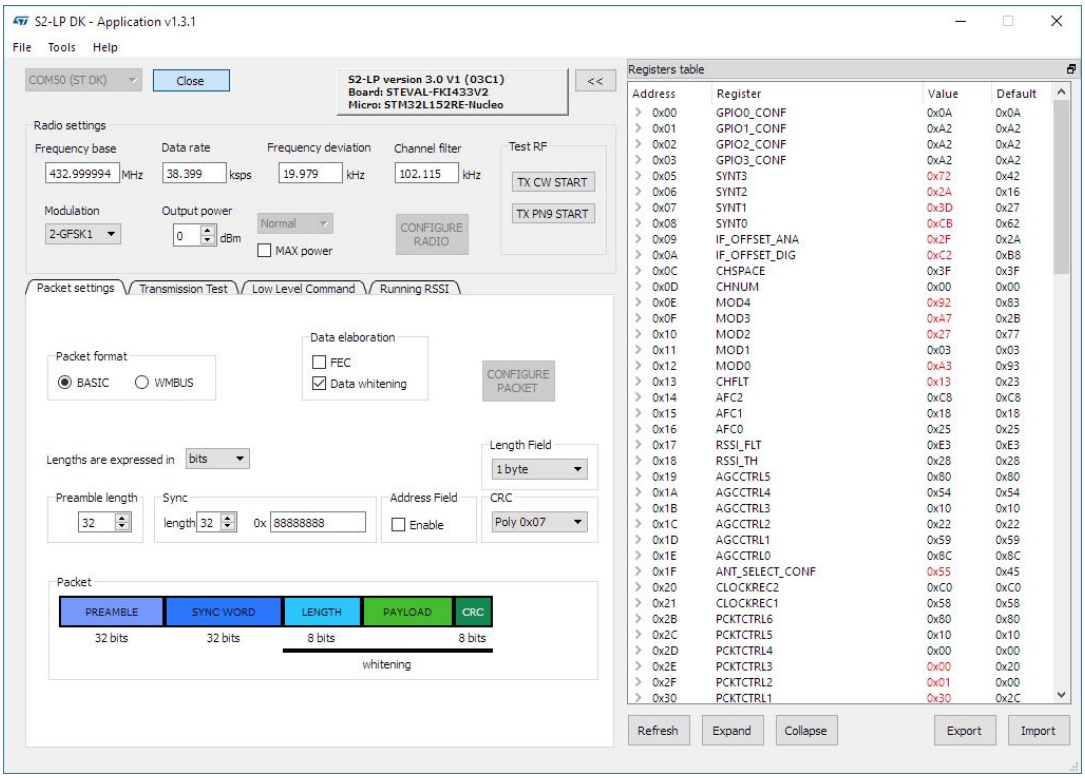
Figure 5. Connection setup 2: 2 PCs with S2-LP-DK GUI

During the tests, each S2-LP DK - DB can work as a transmitter (TX) or a receiver (RX).

The TX device is used as a transmitter during the communication tests; the RX device is used as a receiver during the communication tests. The user can configure the S2-LP DK - DB as a TX device or RX device and dynamically change this selection before running a test.

When the user runs the S2-LP DK.exe file, the S2-LP DK - GUI windows appears as shown below:

Figure 6. S2-LP GUI main window



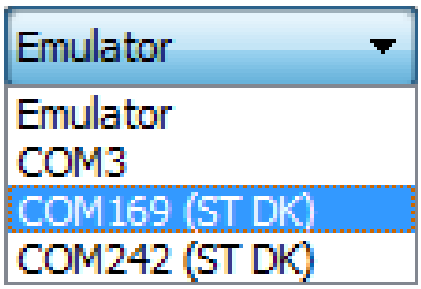
3.2.1 Connection panel

At the top of the main window, the user can select the appropriate available COM port from a drop down list. Once the correct S2-LP COM port is selected, and the open button clicked, the default configuration of S2-LP is loaded and displayed on the S2-LP DK GUI.

Click the COM list also to refresh the available COM port list.

The COM ports associated with the ST development kits are labelled as "(ST DK)".

Figure 7. Available COM ports



3.2.2 Radio setting panel

The radio setting panel is always shown, informing the user about:

- frequency base;
- modulation;

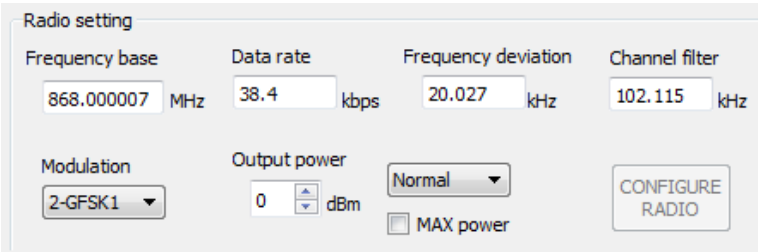
- data rate;
- frequency deviation;
- channel filter
- output power.

These fields can be changed according to these limits (the values may change slightly according to the XTAL frequency):

- frequency base
  - Middle band: [413 - 479] MHz - [452 - 527] MHz
  - High band: [826 - 958] MHz - [904 - 1055] MHz
- modulation:
  - 2-FSK
  - 2-GFSK BT 0.5
  - 2-GFSK BT 1
  - 4-FSK
  - 4-GFSK BT 0.5
  - 4-GFSK BT 1
  - ASK
  - OOK
- data rate interval: [0.3 - 250] kbps.
- frequency deviation interval: [0.793 - 761] kHz.
- channel filter interval: [1.1 – 769.3] kHz.
- output power interval: [-30.0 14.0] dBm if the Normal (without external PA) configuration is selected (as for the STEVAL-FKI433V2, STEVAL-FKI868V2 or X-NUCLEO-S2868A2).
- output power interval: [-5.0 28.0] dBm if the PA configuration is selected (as for the STEVAL-FKI915V1 or X-NUCLEO-S2915A1).

Clicking the "Configure radio" button, all the values are sent to the device and then read and shown.

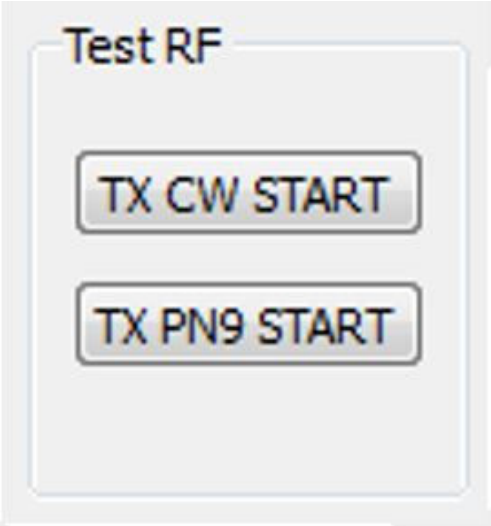
Figure 8. S2-LP radio setting



3.2.3 RF test mode

TX CW and TX PN9 commands put the S2-LP in test mode.

Figure 9. **RF test mode buttons**



Both tests require only one device connected to PC.

3.2.3.1 *TX CW test*

To start this test mode:

1. select the desired radio settings and load them by clicking the "Configure radio" button;
2. Click the "TX CW START" button.

Through this test mode, S2-LP transmits a continuous wave (CW) at the selected frequency and with the selected output power. The user can measure the output signal at the suitable SMA connector or the TX state current consumption.

The S2-LP stays in TX state until the "TX CW STOP" button is clicked.

To change frequency or output power, stop the running test first and then repeat steps 1 and 2 selecting the desired frequency or output power during step 1.

3.2.3.2 *TX PN9 test*

To start this test mode:

1. select the desired radio settings and load them by clicking the "Configure radio" button;
2. click the "TX PN9 START" button.

Through this test mode, S2-LP transmits a PN9 data stream modulated according to the radio setting.

The user can measure the output signal at the suitable SMA connector or the TX state current consumption.

The S2-LP stays in TX state until the "TX PN9 STOP" button is clicked.

To change frequency, output power or modulation scheme, stop the running test first and then repeat steps 1 and 2 selecting the desired frequency, output power or modulation scheme during step 1.

3.3 Packet setting

Select "Panel setting" to view the packet configurations available.

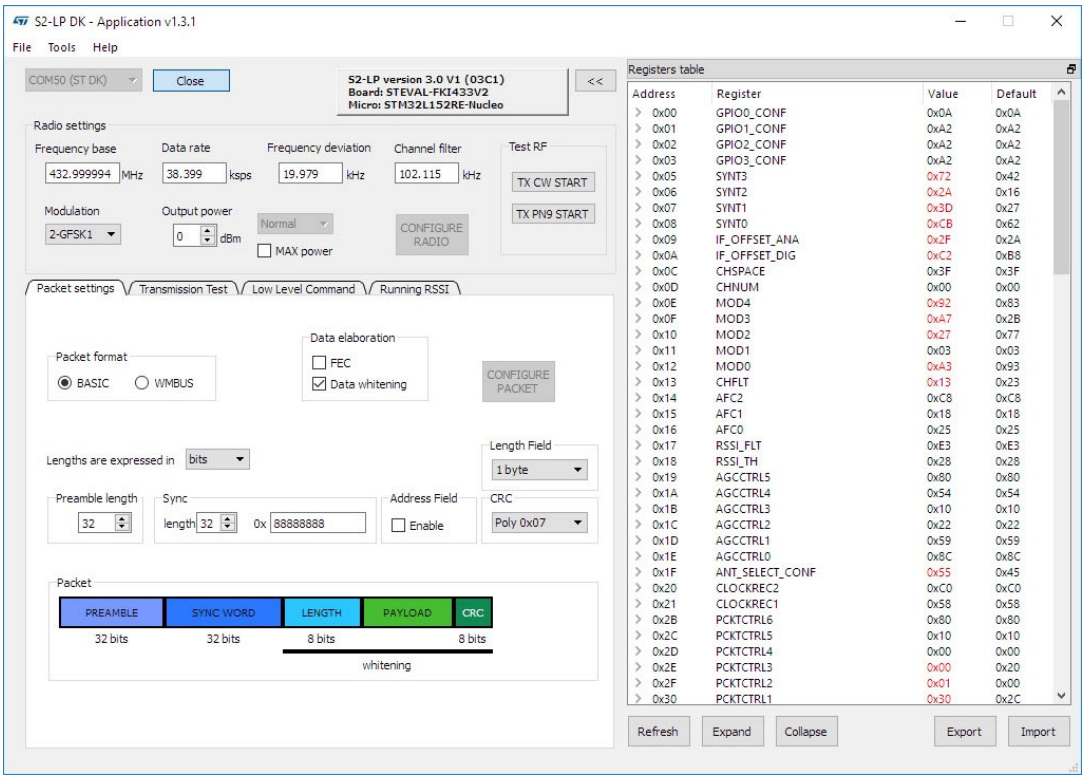
First select the desired packet format by pressing one of the radio buttons in the "Packet format" panel.

The user can choose:

- BASIC
- WMBUS

Each packet format gives different packet setting options.

Figure 10. **Basic packet setting panel**



3.3.1 Packet setting: BASIC

The options for this packet (default configuration) are:

- Preamble length
- Sync length
- Sync value
- CRC
- FEC
- Data whitening

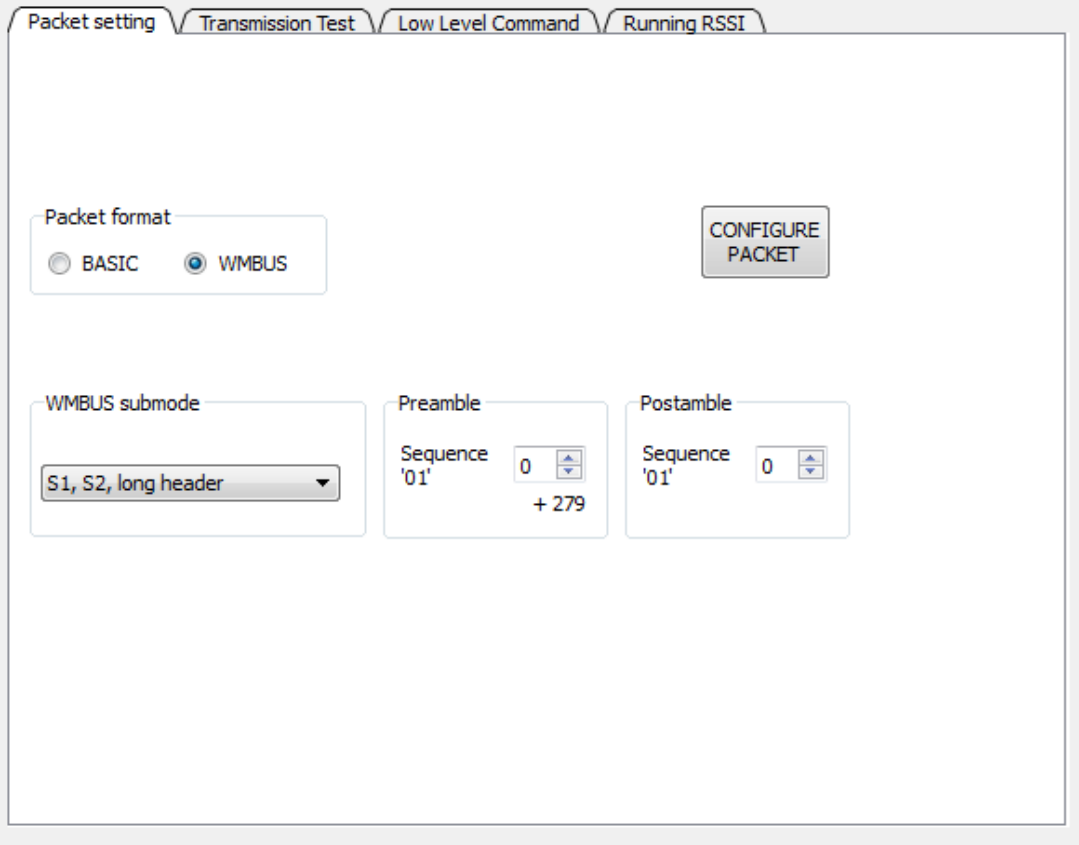
These fields can be changed according to:

- Preamble length interval.
- Sync length interval.
- CRC can be one of the following:
  NO CRC.
  Poly 0x07 (1 byte).
  Poly 0x8005 (2 bytes).
  Poly 0x1021 (2 bytes).
  Poly 0x864CFB (3 bytes).
  Poly 0x04C011BB7 (4 bytes).

The "FEC" and the "Data whitening" can be checked according to the desired setting; if checked, these features are used during the transmission.

3.3.2 Packet setting: WMBUS

Figure 11. **wM-Bus packet setting panel**



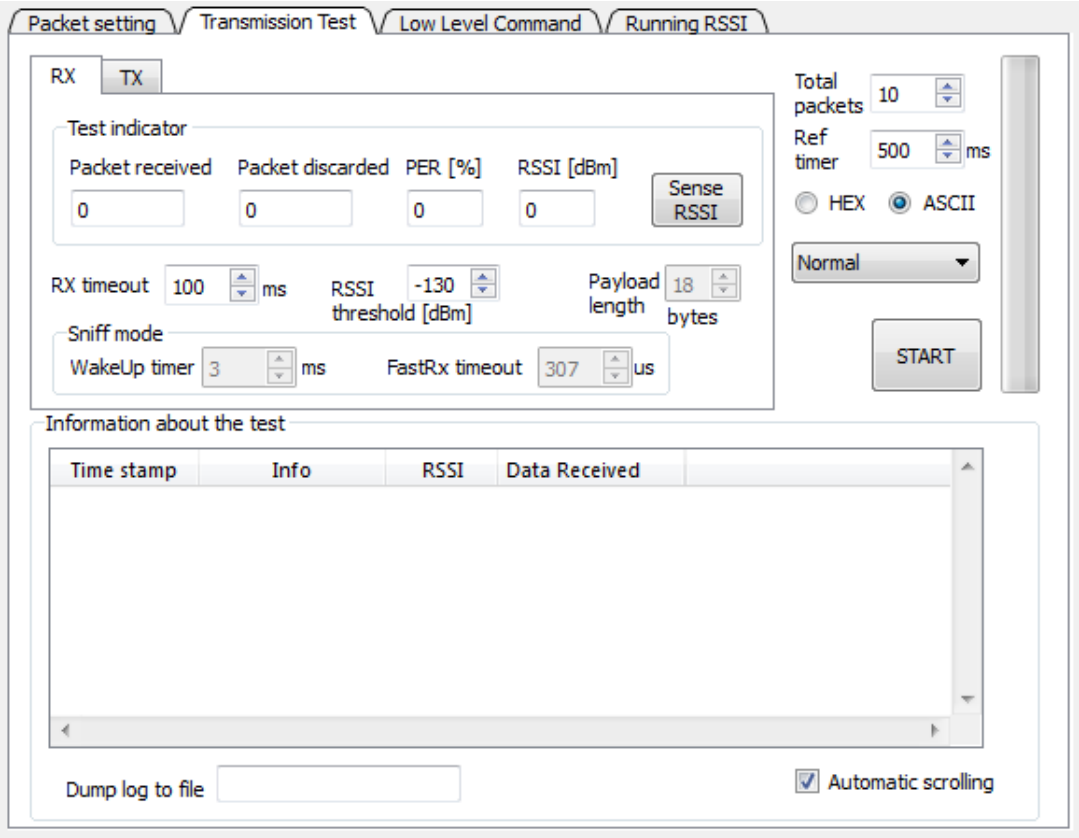
As shown, selecting wM-Bus, S2-LP uses certain parameters for the desired wM-Bus submode.
These fields can be changed according to:

- Preamble length interval: [0 - 1024] chip sequence (01).
- Postamble length interval: [0 - 64] chip sequence (01).
- wM-Bus submode:
  - S1, S2, long header.
  - S1m, S2, T2 other to meter.
  - T1, T2, meter to other.
  - R2 short header.

3.4 Transmission test

Selecting the "Transmission test" view, the user can access all the available packet tests to run the transmission.

Figure 12. **Transmission test panel**



Device role panel

In the "Device role" panel in the left corner, you can set the main role of the device during transmission (RX and TX).

Data to send panel

The Data to send panel has the hexadecimal value or characters sent by a transmitter. The maximum length of this field is 255 bytes (GUI limitation) and represents the effective payload sent. If the HEX check box is selected, the value must be added as 07 08 09 0A and so on; if the ASCII check box is selected, characters are accepted.

It is also possible to generate a random set of bytes by clicking "Random". In this case, the random sequence has a length equal to the one set by the payload length field. Since it is not certain that the randomly generated characters can be converted to ASCII, they are always represented in HEX format.

RX timeout box

In the RX timeout box, the RX timeout in milliseconds should be set to a value large enough to receive the full SYNC word (afterwards, the timer is stopped). It can be set to an approximate value that is larger than the time duration of the preamble and the sync lengths. If the value is 0, the RX timeout is infinite and the S2-LP remains in the RX state until it finds a correct SYNC word.

HEX or ASCII radio buttons

The data received can be displayed in HEX or ASCII format. If ASCII is set and a non-ASCII character is received, the representation automatically switches to HEX format.

Packet length modes

The S2-LP has the following packet length modes:

- variable - the receiver retrieves the packet length from the packet itself (information stored by the transmitter in a field). To set the VAR mode (default), the PCKTCTRL2 register bit 0 must be set to 1.
- fixed - needs the receiver to know the length of the expected packet. To set fix mode, the PCKTCTRL2 register bit must be reset and the GUI disables the length field on the RX tab. This operation can be easily done by the register table. If the fix mode is selected, the RX tab in the transmission test will unlock the payload length field, making it settable by the user.

The Test indicator panel

The Test indicator panel shows all the results about the transmission/reception operations: the packet number correctly received, the packet lost, the RSSI value, the PER since the communication start.

- On the right side, there are these controls:
  - total packets: sets how many packets the transmitter will send or how many packets are expected by the receiver (an infinite number of packets can be set if the value is 0).
  - ref timer: sets the period of time to enter in RX or TX.
- The test defines a cycle in which:
  - if the S2-LP is configured as a transmitter, it sends a packet then the device enters the idle state until the period set in the reference time box expires; then the cycle is repeated. The duration of this operation depends on the data rate and the approximate value is reported in the packet duration box.
  - if the S2-LP is configured as a receiver, the test works in a similar way: the device goes into RX state a couple of milliseconds before the transmitter goes into TX state, then waits for the packet SYNC for the time written in the RX timeout box. If the packet is received or the RX timeout expires, the S2-LP enters the idle state (ready) until the period set in the packet rate box expires. During the first communication, the S2-LP enters the RX state waiting for the first packet (synchronization packet) with infinite RX timeout.
- The RSSI threshold [dBm] sets the RSSI threshold. For good communication, set an RSSI threshold in the receiver greater than the RX noise floor.
- The Sense RSSI button can be used to read the RF power in the air according to the center frequency and the channel filter bandwidth configured. If the Sense RSSI button is clicked when there is no signal in the air, this feature reveals the RF noise in the environment.

Reference time

It is crucial to set the reference time value greater than the packet duration. Otherwise, the received packet can be truncated or not received at all. Furthermore, the packet rate must be the same for both devices.

Start/stop button

The Start button runs the test and turns into a Stop button while the test is stopped.

Low power modes

Normal

In this mode, the device is set to RX or TX by the microcontroller and the idle state (when it is not in RX or TX) is READY. The microcontroller timer is used to implement the reference timer. When this timer expires, the MCU sets the S2-LP in active mode (RX or TX).

LDC

In this mode, the S2-LP is configured with the embedded Low Duty Cycle mode. The device idle state is SLEEP (in RX it is possible to choose the SLEEP A or B by setting the SLEEP_MODE_SEL bit in the PM_CONF0 register).

The wake up event is generated by the embedded WAKE UP timer clocked by the internal low power RC oscillator. The wake up timer value is set equal to the GUI Ref Timer value.

In RX, the automatic reload on the SYNC function is used.

Sniff (RX only)

This configures the device enabling the fast RX termination and the LDC: when the start button is clicked, it starts in the microcontroller (a specific reception routine that manages this mode).

In sniff mode, it is important to set an adequate RSSI threshold to make the receiver work in low-power. The RSSI threshold must be higher than the noise floor seen by the device (otherwise the RSSI threshold is always asserted and thus the fast RX timeout is always stopped).

The wake up timer spin box can be used to configure the time interval between two consecutive RX windows. The fast RX timeout spin box can be used to configure the fast RX timer used to sense the RSSI from the channel. This timer is scaled by the channel filter exponent. Moreover, to set the receiver in the condition of receiving each packet, the wake up timer must be configured to wake up the device, at least twice, inside the preamble. For example, it could be set minor than the preamble duration, which can be computed as the TX (in bits) preamble length divided by the data rate.

For details on the new S2-LP Consumption GUI refer to Section 4 S2-LP consumption tool.

File name textbox

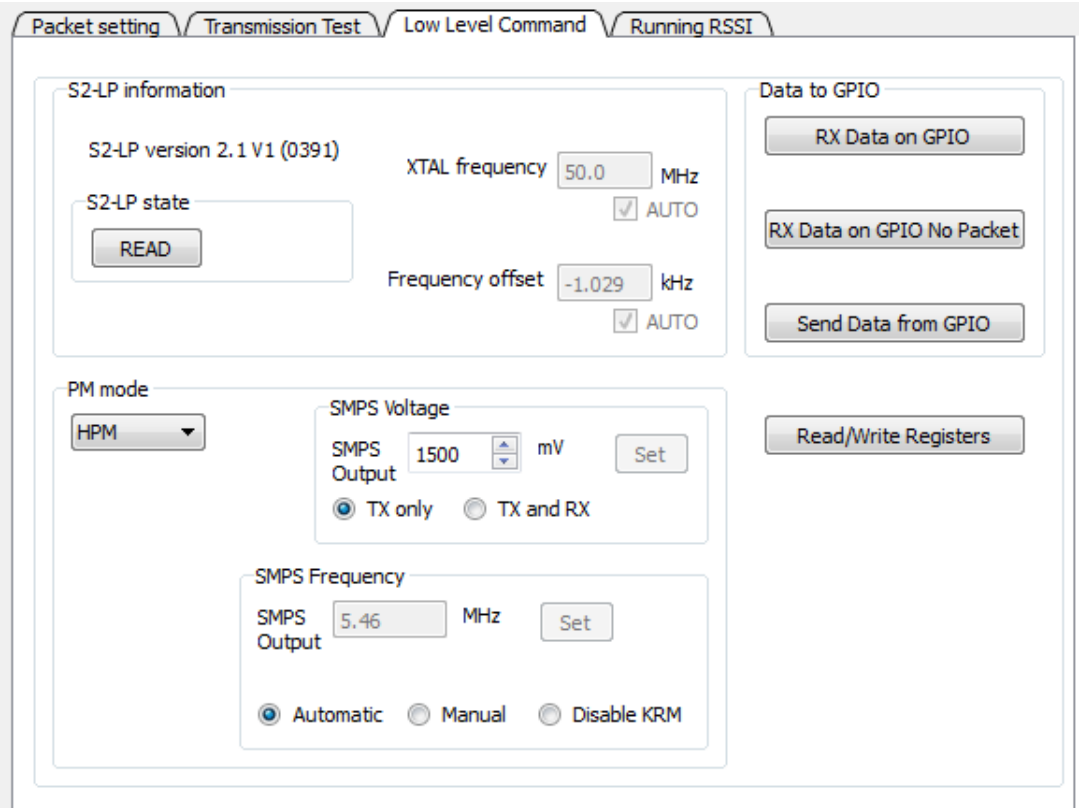
In the transmission test panel bottom, there is a textbox where you can write the file name in which the GUI saves a log of the current test.

Note: This operation is performed during the test, so it is important to write the file name before the test starts.

3.5 Low level commands

Selecting the "Low level commands", you can access the S2-LP test modes, read the status and set the SMPS output voltage.

Figure 13. Low level command panel



The S2-LP status can be read by clicking the S2-LP state read button. The chip version is also shown.

The XTAL frequency is available in the XTAL frequency textbox.

Test modes

Three buttons allow setting some particular test modes:

- RX data in GPIO: it configures the S2-LP GPIO_0 and GPIO_1 to send respectively the RX data received and the clock signal. In this way, when the S2-LP goes into RX state, it is possible to see the received packet.
- RX data in GPIO No Packet: it is equal to the RX data in GPIO with the only difference that the packet handler embedded in the S2-LP is by-passed.
- send data from GPIO: it configures the S2-LP GPIO_0 and GPIO_1 to send respectively the data to transmit and the clock to sample the data. In this way, when the S2-LP goes into the TX state, it is possible to send data loaded through the GPIO (and not through the FIFO).

SMPS voltage

You can set the SMPS voltage (in mV) in the SMPS output box.

SMPS frequency

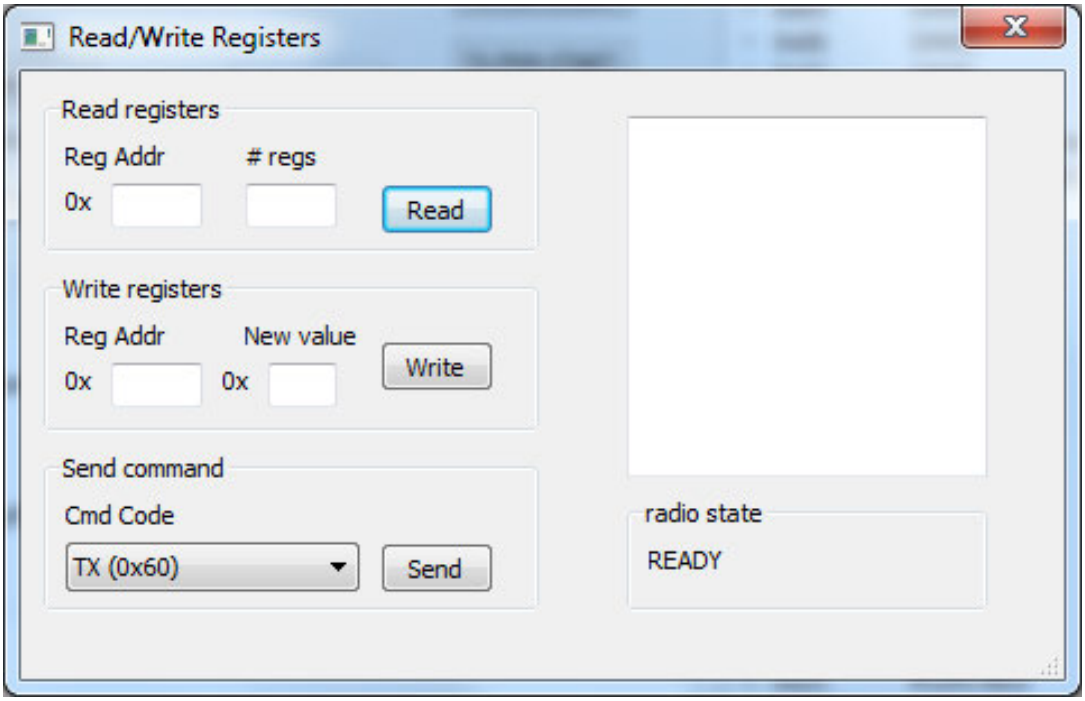
The SMPS frequency can be set in MHz.
The following options are available:

- auto: the optimal frequency is set according to the state of the device (TX or RX);
- manual: the user can specify the SMPS frequency;
- disable KRM: the SMPS frequency is fixed to the value F_dig/4.

Read/Write Registers

While you can write the most used registers through the register table, the Read/Write Registers button allows writing all the S2-LP registers. When clicked, the following window is shown:

Figure 14. Read/write register window



- Through the "Read registers" box, you can specify the starting address and the number of registers to be read from there on.
- Through the "Write registers" box, it is possible to specify the address and the value of a single register.

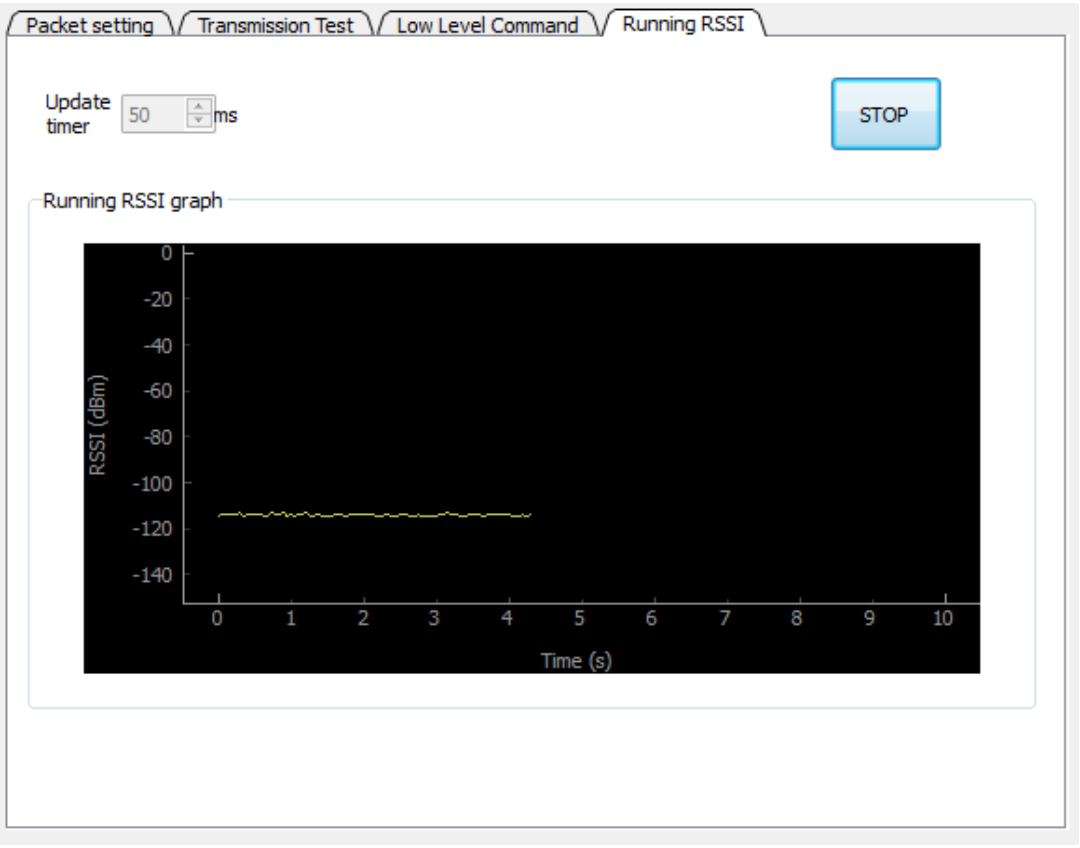
3.6 Running RSSI

The running RSSI tab allows the user to measure the power on the channel using the S2-LP running RSSI feature.

The RSSI values are sampled and plotted in a graph (RSSI (dBm) vs time(s)).

The polling interval is settable by the update timer spin-box.

Figure 15. Running RSSI tab



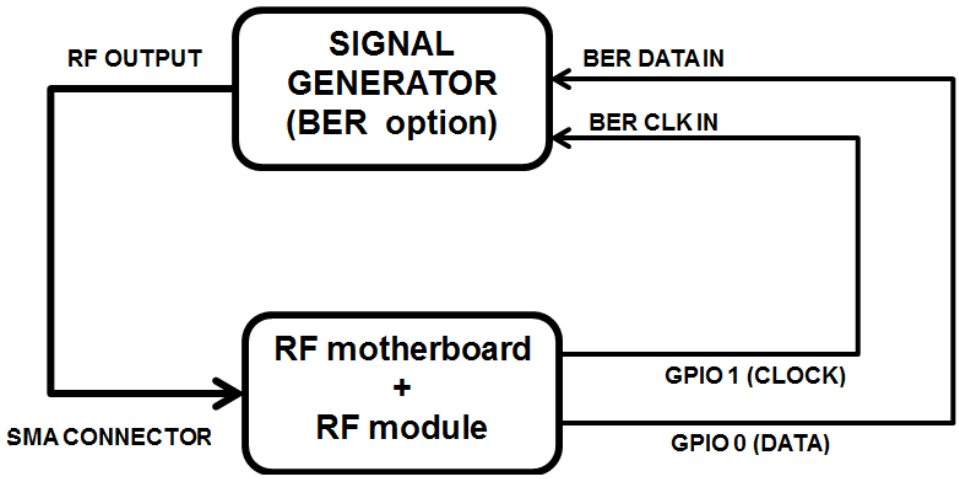
3.7 How to run a BER test using a signal generator

Through the low level command tab, you can put the S2-LP directly in RX mode through GPIOs; the packet handler is therefore totally bypassed and the demodulated data plus associated clock signal is available on two GPIOs.

This mode is enabled by the "RX Data on GPIO No Packet" button. The two signals then can be used in a signal generator with BER option to allow measuring the bit error rate according to that particular radio configuration (see Figure 16. BER test bench schema).

The data must be sampled on the clock signal falling edge.

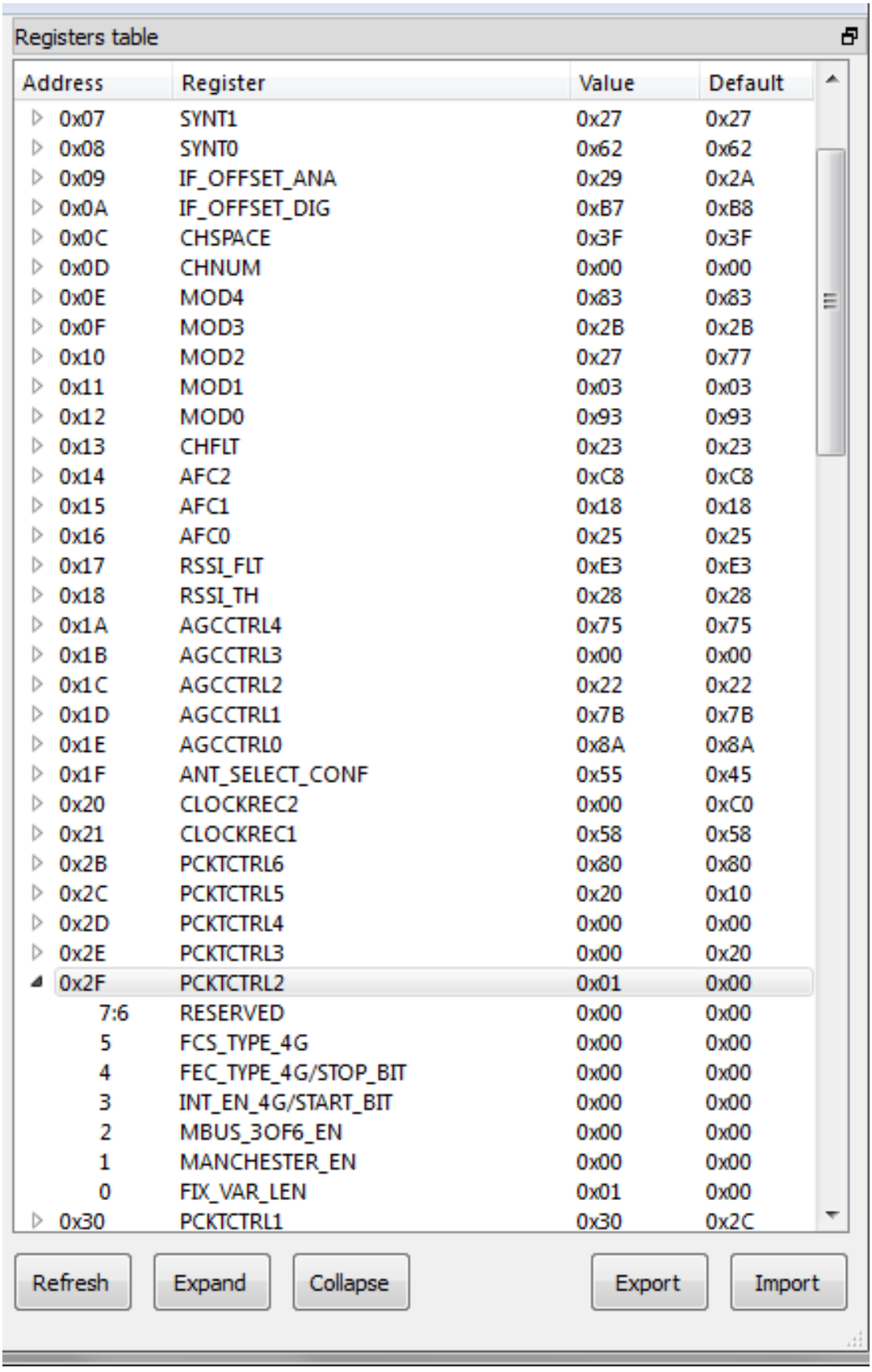
Figure 16. **BER test bench schema**



3.8 Register table

On the right side of the GUI, a register table is shown by default (it can be hidden/shown using the “<<” button). The register table provides a quick and user-friendly way to modify the device registers and bit-fields.

Figure 17. **Register table**



The single register can be expanded or compressed to show its logical fields by clicking on the white arrow to the left of each entry.

When a field is modified, the corresponding register is automatically written in S2-LP.

Moreover, if the register modifies a parameter of the radio part or packet, the corresponding tab is updated with the new field value. It is also possible to get a detailed description of a register by double clicking on its entry in the register table.

Five buttons are available on the tab bottom side:

- Refresh: reads all the register value from the device and updates it into the tab.
- Expand: expands all the registers with the bit-fields.
- Collapse: collapses all the bit-fields.
- Export: saves the register configuration to a file selected by the user.
- Import: loads the registers from a file selected by the user. The file can be loaded both in XML and txt.

3.9 Menu bar

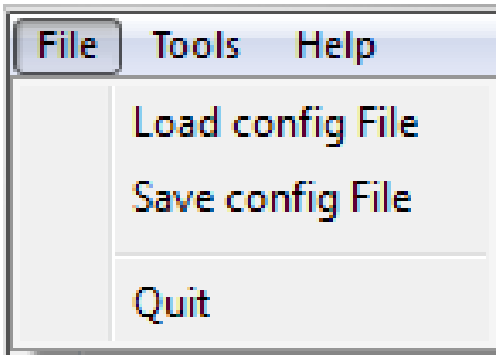
The GUI has a menu bar that exports different functions:

- File
- Tools which includes:
  - firmware upgrade
  - firmware version
  - export code configuration
  - export production info
- Help

3.9.1 File

The file menu provides the following list:

Figure 18. **File list**



3.9.1.1 Save and load high level configurations

The save option allows saving the current radio and packet configurations in a file, so you can reload it easily.

The load option allows loading the stored radio and packet configurations from a file.

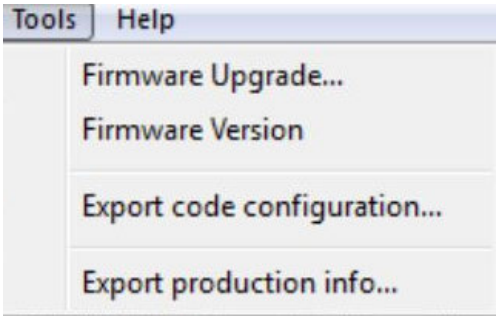
3.9.1.2 Save and load register configurations

These features are available through the export and import buttons below the register table.

3.9.2 Tools

The Tools menu provides the following list:

Figure 19. **Tool list**



3.9.2.1 *Firmware upgrade*

The S2-LP firmware allows performing automatic firmware upgrade via the USB port.

To upgrade the firmware:

1. Launch the S2-LP DK GUI.
2. Select the COM port of the motherboard to upgrade.
3. From Tools→Firmware Upgrade select the firmware image to load (in .bin or .hex format).
4. By clicking OK, the firmware is programmed into the board.

The S2-LP DK Binaries directory contains the image to run the GUI S2LP_CLI_NUCLEO firmware.

Alternatively, it is possible to directly copy the S2LP_CLI_NUCLEO firmware in the hard drive corresponding to the motherboard to be flashed.

If the GUI finds a firmware that is not coherent, it raises a warning and prompts the user to upgrade the firmware.

3.9.2.2 *Firmware version*

The "firmware version" shows the current firmware version running on the microcontroller.

The firmware version format is x.y.z with option BETA to identify beta release and ALPHA to identify alpha release.

Note: A beta release is prior to a final release with the same version number, that is: 2.0.0_BETA is less recent than 2.0.0.

3.9.2.3 *Export code configuration*

This option generates a C-language list of instructions to write new values into the S2-LP registers.

For example, the user can quickly find the device desired configuration using the GUI and then use this tool to obtain a C snippet that can be easily included in the program running on the microcontroller.

Below is an example of the C file obtained through the default configuration:

```
#include <stdint.h>
/**
The SPI interface is platform dependent, this means that it should be implemented according t
o the used hardware.
The function S2LPSpiWriteRegisters(uint8_t address, uint8_t n_regs, uint8_t* buffer) must be
implemented.
An example of implementation (for the SDK_EVAL motherboards of the S2-LP kit) can be found in
to the file: Projects/Drivers/BSP/SDK_Eval_STM32L/Src/SDK_EVAL_Spi_Driver.c
It is advisable to implement also the read registers, the command strobe and the 2 FIFO funct
ions to read and write (no reference in this template code but needed in applications using t
he FIFO of the S2-LP).
**/
/* list of the command codes of S2-LP */
#define COMMAND_TX ((uint8_t)(0x60)) /*!< Start to transmit; valid only from READY */
#define COMMAND_RX ((uint8_t)(0x61)) /*!< Start to receive; valid only from READY */
#define COMMAND_READY ((uint8_t)(0x62)) /*!< Go to READY; valid only from STANDBY or SLEEP or
 LOCK */
#define COMMAND_STANDBY ((uint8_t)(0x63)) /*!< Go to STANDBY; valid only from READY */
#define COMMAND_SLEEP ((uint8_t)(0x64)) /*!< Go to SLEEP; valid only from READY */
#define COMMAND_LOCKRX ((uint8_t)(0x65)) /*!< Go to LOCK state by using the RX configuration
of the synth; valid only from READY */
#define COMMAND_LOCKTX ((uint8_t)(0x66)) /*!< Go to LOCK state by using the TX configuration
of the synth; valid only from READY */
#define COMMAND_SABORT ((uint8_t)(0x67)) /*!< Force exit form TX or RX states and go to READY
 state; valid only from TX or RX */
#define COMMAND_SRES ((uint8_t)(0x70)) /*!< Reset of all digital part, except SPI registers *
/
#define COMMAND_FLUSHRXFIFO ((uint8_t)(0x71)) /*!< Clean the RX FIFO; valid from all states *
/
#define COMMAND_FLUSHTXFIFO ((uint8_t)(0x72)) /*!< Clean the TX FIFO; valid from all states *
/
/* This is the function that initializes the S2-LP with the configuration that the user has e
xported using the GUI */
void SpiritBaseConfiguration(void)
{
 uint8_t tmp[5];
 tmp[0]= 0x92; /* reg. GPIO0_CONF (0x00) */
 tmp[1]= 0x52; /* reg. GPIO1_CONF (0x01) */
 tmp[2]= 0x2A; /* reg. GPIO2_CONF (0x02) */
 S2LPSpiWriteRegisters(0x00, 3, tmp);
 tmp[0]= 0x2B; /* reg. SYNT2 (0x06) */
 tmp[1]= 0x85; /* reg. SYNT1 (0x07) */
 tmp[2]= 0x1F; /* reg. SYNT0 (0x08) */
 tmp[3]= 0x2F; /* reg. IF_OFFSET_ANA (0x09) */
 tmp[4]= 0xC2; /* reg. IF_OFFSET_DIG (0x0A) */
 S2LPSpiWriteRegisters(0x06, 5, tmp);
 tmp[0]= 0x92; /* reg. MOD4 (0x0E) */
 tmp[1]= 0xA7; /* reg. MOD3 (0x0F) */
 tmp[2]= 0x27; /* reg. MOD2 (0x10) */
 S2LPSpiWriteRegisters(0x0E, 3, tmp);
 tmp[0]= 0xA3; /* reg. MOD0 (0x12) */
 tmp[1]= 0x13; /* reg. CHFLT (0x13) */
 S2LPSpiWriteRegisters(0x12, 2, tmp);
 tmp[0]= 0x55; /* reg. ANT_SELECT_CONF (0x1F) */
 tmp[1]= 0x00; /* reg. CLOCKREC2 (0x20) */
 S2LPSpiWriteRegisters(0x1F, 2, tmp);
 tmp[0]= 0x20; /* reg. PCKTCTRL5 (0x2C) */
 S2LPSpiWriteRegisters(0x2C, 1, tmp);
 tmp[0]= 0x00; /* reg. PCKTCTRL3 (0x2E) */
 tmp[1]= 0x01; /* reg. PCKTCTRL2 (0x2F) */
 tmp[2]= 0x30; /* reg. PCKTCTRL1 (0x30) */
 S2LPSpiWriteRegisters(0x2E, 3, tmp);
 tmp[0]= 0x01; /* reg. PROTOCOL1 (0x3A) */
 S2LPSpiWriteRegisters(0x3A, 1, tmp);
 tmp[0]= 0x41; /* reg. PCKT_FLT_OPTIONS (0x40) */
 S2LPSpiWriteRegisters(0x40, 1, tmp);
 tmp[0]= 0x00; /* reg. FAST_RX_TIMER (0x54) */
 S2LPSpiWriteRegisters(0x54, 1, tmp);
```

```
 tmp[0]= 0x1D; /* reg. PA_POWER8 (0x5A) */
 S2LPSpiWriteRegisters(0x5A, 1, tmp);
 tmp[0]= 0x07; /* reg. PA_POWER0 (0x62) */
 tmp[1]= 0x01; /* reg. PA_CONFIG1 (0x63) */
 S2LPSpiWriteRegisters(0x62, 2, tmp);
}
```

According to the application, it can be manually modified to become a macro or a simple instruction block.

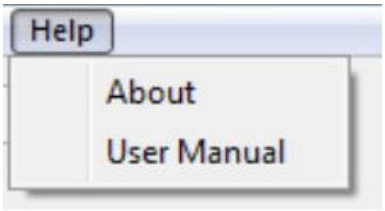
3.9.2.4 *Export production info*

Each STEVAL-FKI board is equipped with a E2PROM with some data stored during the manufacturing phase that can be exported to a file by clicking on "Export production info".

3.9.3 Help

The Help button gives you access to the user manual.

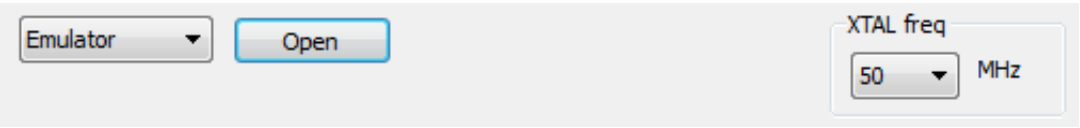
Figure 20. Help menu



3.9.4 Device emulator

It is possible to simulate the device without any board connected to the PC through the "Emulator" button: when selected, the user can use the GUI as if a board were connected to the PC.

Figure 21. S2-LP emulator



Since there is no actual board, the user should manually specify an XTAL frequency (otherwise automatically computed by the microcontroller) using the tab which only becomes active in this case.

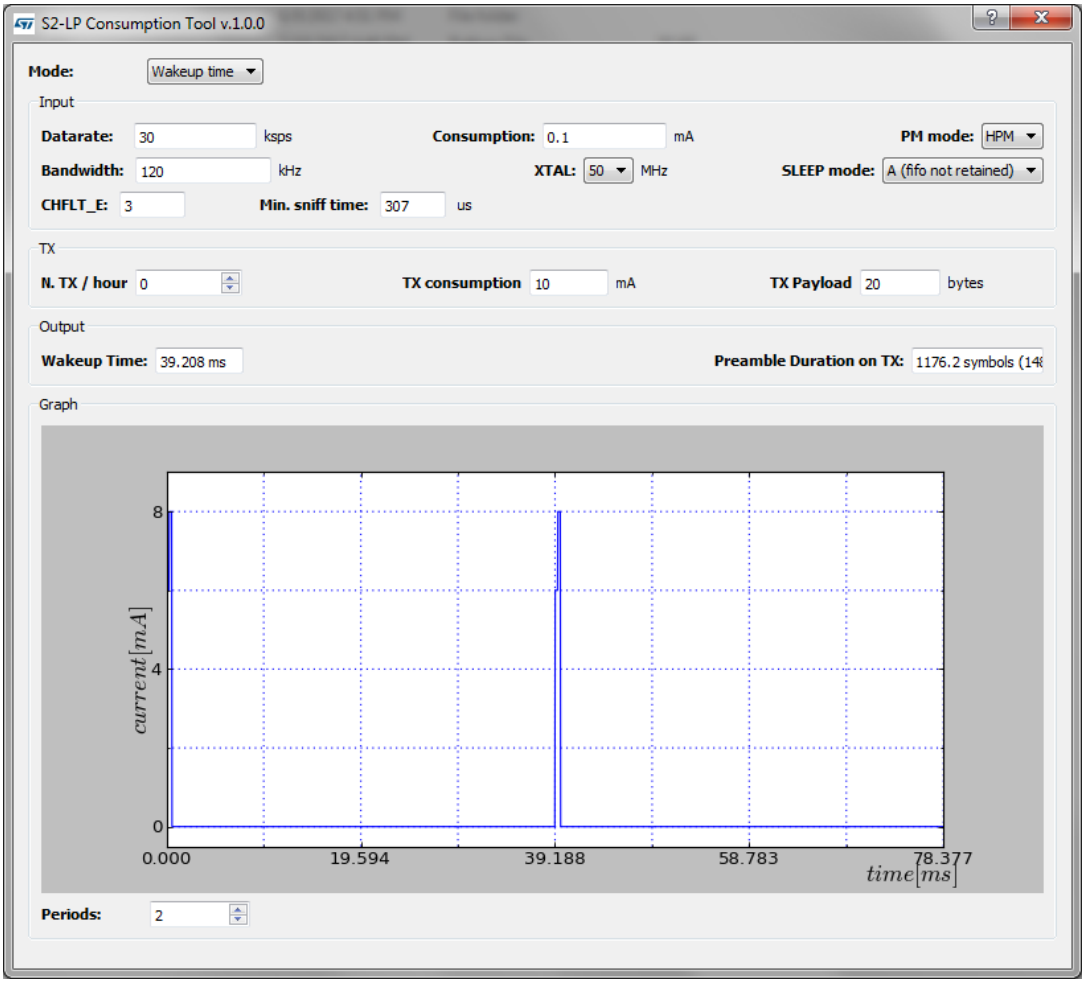
Subsequently, clicking the "Open" button, everything related to the device configuration should run exactly as if a device were connected.

The user can therefore easily select his own configuration and see or save the register values needed to keep the same configuration on his firmware. For this purpose, this feature can be used in cooperation with the "Export code configuration" tool.

4 S2-LP consumption tool

The S2-LP Consumption Tool is a graphical user interface to simulate the S2-LP current consumption in sniff mode. As it constitutes a simulation, it has not to be considered as a replacement of measurement on the actual silicon which gives the actual figures.

Figure 22. S2-LP consumption tool - GUI overview



The GUI can be set in two modes:

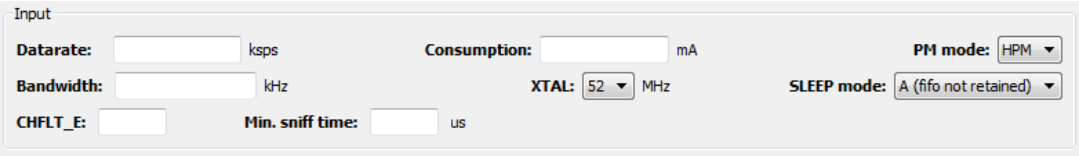
- Wakeup
- Consumption

4.1 Wakeup mode

In **Wakeup** mode, the GUI role is computing the wakeup time needed to reach a desired consumption.
The input parameters are:

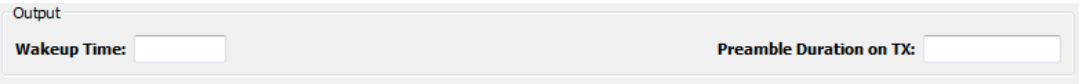
- the desired communication **Datarate**
- the target **Consumption**
- the channel filter**Bandwidth**
- the **XTAL** frequency
- the **PM Mode** (HPM/LPM)
- the **SLEEP mode** (A or B)

Figure 23. **Wakeup mode input**



The output is the wakeup time and the preamble minimum length on the TX to ensure each packet is correctly received.

Figure 24. **Wakeup mode output**

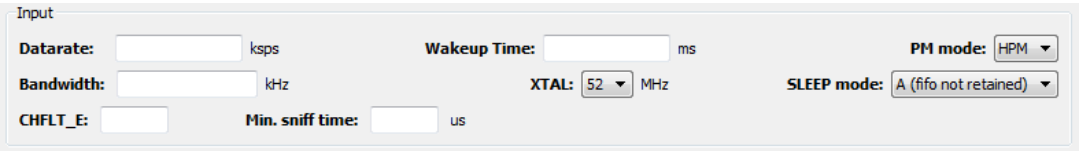


4.2 Consumption mode

In **Consumption** mode, the GUI computes the consumption given the wakeup time as input data.
The input parameters are:

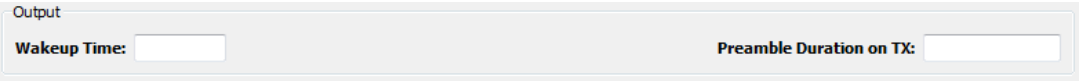
- the desired communication **Datarate**
- the **Wakeup time**
- the channel filter**Bandwidth**
- the **XTAL** frequency
- the **PM Mode** (HPM/LPM)
- the **SLEEP mode** (A or B)

Figure 25. **Consumption mode input**



The output is the current consumption and battery duration indication.

Figure 26. **Consumption mode output**

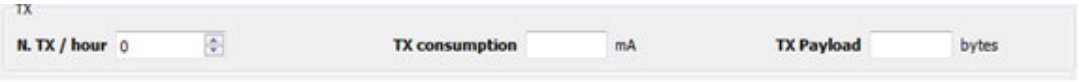


For both modes the GUI performs the computation once all the input fields are filled in.

4.3 TX

The GUI also takes into account the consumption due to transmission of packets.
From the TX section, it is possible to specify the number of transmissions per hour, the TX current and the TX payload in bytes.

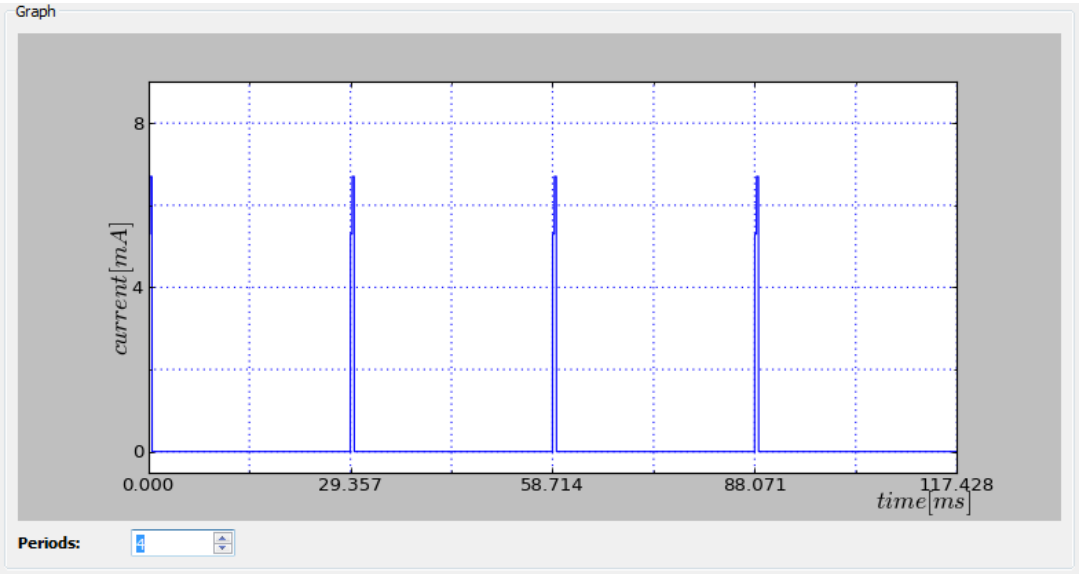
Figure 27. **TX section**



4.4 Current profile graph

The following picture shows the wakeup RX cycles.

Figure 28. Current graph



5 Firmware examples

5.1 Command line interface (CLI)

This S2-LP CLI demo firmware shows how to use a command line interface to send commands to the S2-LP from different MCUs. The same firmware is used by the GUI to drive the S2-LP and perform evaluation procedures.

To use the CLI without the GUI, follow the steps below.

Step 1. Open the COM port with a baudrate of 115200 8-N-1.
A simple command shell opens.

Step 2. Type 'help' to read the entire list of commands.

Step 3. Type the name of the desired command followed by parameters (if necessary).

5.1.1 CLI commands, modulation types and S2-LP modes

Table 4. CLI commands

Command	Arguments	Description
`S2LPRadioInit`	`modulation_select`: the type of modulation(1) `data_rate`: data rate in sym/s `fdev`: frequency deviation in Hz `rx_bandwidth`: rx filter bandwidth in Hz `xtal_frequency`: suggested value is 0 (in this case the preset XTAL value is used)	Initializes the S2-LP radio
`S2LPRadioGetInfo`	None	Gets radio parameters
`S2LPRadioGetXtalFrequency`	None	Gets the crystal frequency (in Hz)
`S2LPRadioSetFrequencyBase`	`frequency_base`: base frequency in Hz.	Initializes the S2-LP radio base frequency
`S2LPRadioGetFrequencyBase`	None	Gets the radio base frequency (in Hz)
`S2LPRadioSetModulation`	`modulation_scheme`: the type of modulation (1)	Initializes the S2-LP modulation scheme
`S2LPRadioGetModulation`	None	Gets the S2-LP modulation scheme
`S2LPRadioSetPALeveldBm`	`pa_level`: float indicating the dBm value `pa_index`: index of the PA register from 0 to 7	Sets the S2-LP PA power level and the passed index as max index
`S2LPRadioGetPALeveldBm`	`pa_index`: index of the PA register from 0 to 7	Gets the S2-LP PA level in dBm
`S2LPRadioSetPALevelNdBm`	`pa_level`: float indicating the dBm value `pa_index`: index of the PA register from 0 to 7	Sets the S2-LP PA level

Command	Arguments	Description
S2LPRadioSetPALevelMaxInd	pa_index: index of the PA register from 0 to 7	Sets the S2-LP PA max index
S2LPGetPktFrmt	None	Returns the packet format
S2LPPktBasicInit	preamble_length: preamble length in bits sync_length: sync length in bits sync_word: sync word fix_var_length: (0: fixed, 1: var) extended_length: only for variable length (0: 1byte address, 1: 2bytes address) crc_mode(2) address: specifies if address must be used (1) or not (0) fec: specifies if FEC must be used (1) or not (0) whitening: specifies if whitening must be used (1) or not (0)	Initializes the Basic packet format
S2LPPktBasicGetInfo	None	Returns the BASIC packet info
S2LPPktStackInit	preamble_length: preamble length in bits sync_length: sync length in bits sync_word: sync word fix_var_length: (0: fixed, 1: var) extended_length: only for variable length(0: 1byte address, 1: 2bytes address) crc_mode(2) fec: specifies if FEC must be used (1) or not (0) whitening: specifies if whitening must be used (1) or not (0)	Initializes the STack packet format
S2LPPktStackGetInfo	None	Returns the STack packet info
S2LPPktBasicSetPayloadLength	payload_len: payload length to be set	Sets the payload length for Basic packets
S2LPPktBasicGetPayloadLength	None	Gets the payload length for Basic packets
S2LPPktStackSetPayloadLength	payload_len: payload length to be set	Sets the payload length for STack packets

Command	Arguments	Description
S2LPPktStackGetPayloadLength	None	Gets the payload length for STack packets
S2LPPktMbusSetPayloadLength	payload_len: payload length to be set	Sets the payload length for wM-Bus packets
S2LPPktMbusGetPayloadLength	None	Gets the payload length for STack packets
S2LPPktMbusInit	wmbus_submode[3] preamble_length: preamble length in chips postamble_length: preamble length in chips	Initializes the wM-Bus packet format
S2LPPktMbusGetInfo	None	Returns the wM-Bus packet info
S2LPTimerSetRxTimeoutUs	rx_timeout: RX timeout in µs	Sets the RX timeout
S2LPTimerGetRxTimeout	None	Gets the RX tiemout
S2LPGpioInit	gpio_pin: 0: GPIO0, 1: GPIO1, 2: GPIO2, 3: GPIO3 gpio_mode[4] gpio_io function[5]	Sets the S2-LP GPIOs
S2LPIrq	irq code[6] en: 1: enable, 0: disable	Sets the S2-LP GPIOs
S2LPIrqGetStatus	None	Gets the S2-LP IRQ status
S2LPQiGetRssidBm	None	Gets the RSSI dBm value
S2LPGetVersion	None	Gets the S2-LP version (silicon cut and ver)
S2LPWhitening	whitening: 1: enable, 0: disable	Sets the S2-LP whitening
S2LPDirectRfSetTxMode	tx_mode[7]	Sets the S2-LP DirectRf mode in TX
S2LPDirectRfSetRxMode	rx_mode[8].	Sets the S2-LP DirectRf mode in RX
S2LPLinearFifoSetAlmostFullThresholdRx	ae_thr: FIFO almost full threshold	Sets the RX almost full threshold
S2LPLinearFifoSetAlmostEmptyThresholdTx	ae_thr: FIFO almost empty threshold	Sets the TX almost empty threshold
S2LPTimerSetWakeUpTimerUs	wake_up: timer in us	Sets the wake-up timer
S2LPTimerLdcrMode	en: 1: enable, 0: disable en_autoreload: 1: enable, 0: disable	Enables the LDCR mode
S2LPCalibrateRco	None	Calibrates the RCO
S2LPGetNBytesReportAll	en: 1: enable, 0: disable	Reports all the packet data
S2LPGetNBytes	n_bytes: number of bytes to be received	Gets n-bytes routine. It receives the desired number of bytes
S2LPGetNBytesBatch	ref_timer: data will be sent every ref_timer ms.	Get n-bytes batch routine. It receives the desired number of packets

Command	Arguments	Description
	num_packets: the number of times the packets must be sent	
S2LPSendPattern	buffer: data to be sent n_bytes: number of bytes	Sends a buffer in circular mode (suitable to test the polar mode)
S2LPSendPatternRaw	raw_buffer: a byte raw stream	Sends a buffer in a circular way (suitable to test the polar mode)
S2LPSendNBytes	buffer: data to be send through FIFO	Sends n-bytes according to the configured FIFO TX mode (direct, packet, etc.)
S2LPSendNBytesBatch	ref_timer: data are sent every ref_timer ms num_packets: the number of times the packets must be sent buffer: data to be sent through FIFO	Send n-bytes in batch mode according to the configured FIFO TX mode (direct, packet, etc.)
S2LPGetRssiRunBatch	ref_timer: RSSI is read for ref_timer ms	Returns the RSSI value
S2LPSendPatternGpio	pattern: pattern to be sent	Sends a pattern (4 bytes) using the TX through GPIO mode
SdkEvalIrqHandler	blocking: if 1, it is blocked until the interrupt is raised (or stop cmd comes), if 0 it is not blocking.	Waits for an S2-LP IRQ raised and returns the IRQ_STATUS when raised. If other interrupts are in the IRQ queue it pops and returns the last one.
S2LPGetLibVersion	None	Gets the S2-LP library version
S2LPSendBatchLP	en: 1: enable, 0: disable	This causes the low power standby mode to be used in the SendNbytesBatch routines
S2LPGetBatchLP	en: 1: enable, 0: disable	This causes the low power standby mode to be used in the GetNbytesBatch routines
S2LPGetRcoFrequency	None	Gets RCO frequency
SdkEvalSpiCommandStrobes	command_code(9)	Sends a command to the S2-LP
SdkEvalSpiReadRegisters	address: register address n_regs: number of registers to read	Reads register(s) from the device
SdkEvalSpiWriteRegisters	address: register address values: values to write as a block	Writes register(s) to the device
SdkEvalSpiReadFifo	n_bytes: number of bytes to read from the RX FIFO	Reads the RX FIFO
SdkEvalSpiWriteFifo	values: values to write to FIFO as a block	Writes the TX FIFO

Command	Arguments	Description
S2MGpioIrqConfiguration	gpio_pin: 0: GPIO0, 1: GPIO1, 2: GPIO2, 3: GPIO3 en: 1: enable, 0: disable	Configures a GPIO of the uC in EXTI (external interrupt) mode
S2MGpioGetValue	gpio_pin: 0: GPIO0, 1: GPIO1, 2: GPIO2, 3: GPIO3	Gets the value of a GPIO configured as input
SdkEvalSdn	Value: put in SDN (1, pin high), exit from SDN (0, pin low)	Drives the S2-LP shutdown pin
SdkEvalRfboardIdentification	xtal: if this value is 0, the XTAL is automatically detected, otherwise it is set to this value	Identifies the RF board getting some info from the EEPROM and configuring the library
FEMSetBypass	bypass: 0: Bypass NO, 1: Bypass YES	Sets the FEM in bypass mode (where available)
CliGetTimer	None	Gets the CLI timer
CliResetTimer	None	Resets the CLI timer
SdkEvalLedHandler	led: 0: LED1, 1: LED2, 2: LED3, 3: LED4, 4: LED5	Sets a LED on or off (according to the motherboard configuration)
EepromStatus	None	Get the EEPROM status value
EepromWritePage	page: page number offset: offset to start to write from inside the page values: list of the values to write	Writes a value or a list of values in the EEPROM
EepromReadPage	page: page number offset: offset to start reading from inside the page n_bytes: number of bytes to read	Reads a value or a list of values from the EEPROM
SdkEvalGetVersion	None	Gets the motherboard version
TimeMeasureService	gpio_pin: 0: GPIO0, 1: GPIO1, 2: GPIO2, 3: GPIO3	Enables the time measure service on a GPIO
GetTimeMeasure	gpio_pin: 0: GPIO0, 1: GPIO1, 2: GPIO2, 3: GPIO3	Gets the time measurements taken so far if the TimeMeasureServiceAction was enabled
SdkEvalIrqRaised	blocking: if 0 it returns immediately	Waits for an interrupt from the device. Returns when the IRQ is raised from the GPIO. This does not read the IRQ_STATUS registers
SdkEvalHiZ	None	Sets the GPIO and the SPI in high impedance configuration
SdkComBaudrate	baudrate: value in bps	Sets the serial baudrate

1. *Refer to Table 5. Modulation types.*
2. *Refer to Table 12. CRC modes.*
3. *Refer to Table 13. WMBus sub-modes*

4. Refer to Table 6. S2-LP GPIO modes.
5. Refer to Table 7. S2-LP GPIO I/O functions.
6. Refer to Table 8. S2-LP IRQ.
7. Refer to Table 10. Direct TX modes
8. Refer to Table 9. Direct RX modes
9. Refer to Table 11. S2-LP commands.

Table 5. Modulation types

Modulation type	Value
2-FSK modulation	0x00
4-FSK modulation	0x10
2GFSK modulation selected with BT = 0.5	0xA0
2GFSK modulation selected with BT = 1	0x20
G4FSK modulation selected with BT = 0.5	0xB0
G4FSK modulation selected with BT = 1	0x30
OOK modulation	0x50
Polar mode	0x60
No modulation (continuous wave)	0x70

Table 6. S2-LP GPIO modes

GPIO mode	Value
Digital input on GPIO	1
Digital output on GPIO (low current)	2
Digital output on GPIO (high current)	3

Table 7. S2-LP GPIO I/O functions

Input/output configuration	GPIO mode	Value
If configured as output	nIRQ (Interrupt Request, active low), default configuration after POR	0x00
	POR inverted (active low)	0x08
	Wake-Up Timer expiration: "1" when WUT has expired	0x10
	Low battery detection: "1" when battery is below threshold setting	0x18
	TX data internal clock output (TX data are sampled on the rising edge of it)	0x20
	TX state indication: "1" when S2LP1 is passing in the TX state	0x28
	TX/RX FIFO Almost Empty Fla	0x30
	TX/RX FIFO Almost Full Flag	0x38
	RX data output	0x40
	RX clock output (recovered from received data)	0x48
	RX state indication: "1" when demodulator is ON	0x50
	VDD when the device is not in SLEEP or STANDBY	0x58
	VDD when device is in STANDBY	0x60
	Antenna switch used for antenna diversity	0x68

Input/output configuration	GPIO mode	Value
If configured as output	Valid Preamble Detected Flag	0x70
	Sync WordSync Word Detected Flag	0x78
	RSSI above threshold	0x80
	MCU Clock	0x88
	TX or RX mode indicator (to enable an external range extender)	0x90
	VDD (to emulate an additional GPIO of the MCU, programmable by SPI)	0x98
	GND (to emulate an additional GPIO of the MCU, programmable by SPI)	0xA0
	External SMPS enable signal (active high)	0xA8
	Device in SLEEP (active high)	0xB0
	Device in READY (active high)	0xB8
	Device in LOCK (active high)	0xC0
	Device waiting for LOCK (active high)	0xC8
	TX_DATA_OOK signal (internal control signal generated in the OOK analog smooth mode)	0cD0
	Device waiting for a high level of the READY2 signal from XO	0xD8
	Device waiting for timer expiration to allow PM block settling	0xE0
	Device waiting for end of VCO calibration	0xE8
	Device enables the full circuitry of the SYNTH block	0xF0
If configured as input	TX Command	0x00
	RX Command	0x08
	TX data input for direct modulation	0x10
	Wake-up from external input	0x18
	External clock at 34.7 kHz (used for LDC modes timing)	0x20

Table 8. S2-LP IRQ

IRQ mode	Value
IRQ: RX data ready	0x00000001
RX data discarded (upon filtering)	0x00000002
TX data sent	0x00000004
Max re-TX reached	0x00000008
CRC error	0x00000010
TX FIFO underflow/overflow error	0x00000020
RX FIFO underflow/overflow error	0x00000040
TX FIFO almost full	0x00000080
TX FIFO almost empty	0x00000100
RX FIFO almost full	0x00000200
RX FIFO almost empty	0x00000400
Max number of back-off during CCA	0x00000800
Valid preamble detected	0x00001000
Sync word detected	0x00002000

IRQ mode	Value
RSSI above threshold	0x00002000
Wake-up timeout in LDC mode	0x00008000
READY state	0x00010000
STANDBY state after MCU_CK_CONF_CLOCK_TAIL_X clock cycles	0x00020000
Battery level below threshold	0x00040000
Power on reset	0x00080000
Brown out event (both accurate and inaccurate)	0x00100000
LOCK state	0x00200000
RX operation timeout	0x10000000
RX sniff operation timeout	0x20000000
All IRQ	0x7FFFFFFF

Table 9. Direct RX modes

RX mode	Value
Normal RX mode	0x00
Direct RX FIFO mode	0x10
Direct RX GPIO mode	0x20

Table 10. Direct TX modes

TX mode	Value
Normal TX mode	0x00
Direct TX FIFO mode	0x04
Direct TX GPIO mode	0x08
PN9 TX mode	0x0C

Table 11. S2-LP commands

Command	Value
CMD_TX - Start to transmit; valid only from READY	0x60
CMD_RX - Start to receive; valid only from READY	0x61
CMD_READY - Go to READY; valid only from STANDBY or SLEEP or LOCK	0x62
CMD_STANDBY - Go to STANDBY; valid only from READY	0x63
CMD_SLEEP - Go to SLEEP; valid only from READY	0x64
CMD_LOCKRX - Go to LOCK state by using the RX configuration of the synth; valid only from READY	0x65
CMD_LOCKTX - Go to LOCK state by using the TX configuration of the synth; valid only from READY	0x66
CMD_SABORT - Force exit form TX or RX states and go to READY state; valid only from TX or RX	0x67
CMD_LDC_RELOAD - LDC Mode: Reload the LDC timer with the value stored in the LDC_PRESCALER / COUNTER registers; valid from all states	0x68
CMD_RCO_CALIB - Start (or re-start) the RCO calibration	0x69

Command	Value
CMD_SRES - Reset of all digital part, except SPI registers	0x70
CMD_FLUSHRXFIFO - Clean the RX FIFO; valid from all states	0x71
CMD_FLUSHTXFIFO - Clean the TX FIFO; valid from all states	0x72
MD_SEQUENCE_UPDATE - Autoretransmission: Reload the Packet sequence counter with the value stored in the PROTOCOL[2] register valid from all states	0x73

Table 12. CRC modes

CRC mode	Value
No CRC	0x00
Poly 0x07	0x20
Poly 0x8005	0x40
Poly 0x1021	0x60
Poly 0x864CFB	0x80
Poly 0x04C011BB7	0xA0

Table 13. WMBus sub-modes

WMBus mode	Value
WMBUS_SUBMODE_NOT_CONFIGURED	0
WMBUS_SUBMODE_S1_S2_LONG_HEADER	1
WMBUS_SUBMODE_S1_M_S2_T2_OTHER_TO_METER	2
WMBUS_SUBMODE_T1_T2_METER_TO_OTHER	3
WMBUS_SUBMODE_R2_SHORT_HEADER	4

5.1.2 IAR project

The workspace file is called CLI_Project.eww and is placed in the directory Projects/Projects_Cube/S2-LP_DK/S2LP_CLI_Project in the EWARM folder under STM32 or BlueNRG folder.

To use the project with IAR Embedded Workbench for ARM:

Step 1. Open the Embedded Workbench for ARM and select [**File**]>[**Open**]>[**Workspace menu**].

Step 2. Open the IAR project .../ Projects/Projects_Cube/S2-LP_DK/S2LP_CLI_Project and select the EWARM/CLI_Project.eww file under STM32 or BlueNRG folder

Step 3. Select the desired configuration to build

Step 4. Select the download and debug button to recompile and link the entire application, download the related binary image and go in debug mode.

5.1.3 MDK-ARM KEIL project

The workspace file is called CLI_Project.uvprojx and is placed in the directory Projects/Projects_Cube/S2-LP_DK/S2LP_CLI_Project in the MDK-ARM folder under STM32 or BlueNRG folder.

To use the project with KEIL uVision for ARM:

Step 1. Open the KEIL uVision for ARM and select [**Project**]>[**Open Project**] menu

Step 2. Open the KEIL project .../Projects/Projects_Cube/S2-LP_DK/S2LP_CLI_Project and select the MDK-ARM/CLI_Project.uvprojx file under STM32 or BlueNRG folder

Step 3. Select the desired configuration to build

Step 4. Select [**Project**]>[**Rebuild all target files**] to recompile and link the entire application.

Step 5. Select [**Project**]>[**Download**] to download the related binary image.

5.2 Library examples

The S2-LP development kit contains a set of some simple examples showing how to use some S2-LP features (such as the packet handler, low-power modes, auto-retransmissions, CSMA engine, etc).

Each example consists of at least two programs called A and B acting as transmitter and receiver, respectively:

- Basic generic: exchange of S2-LP BASIC packets with a payload length below the FIFO size (128 bytes).
- FIFO handler: exchange of S2-LP BASIC packets with a payload length bigger than the FIFO size.
- wM-Bus STD: exchange of wM-Bus like formatted packets.
- Sniff: exchange of packets using the fast RX termination feature for the receiver.
- LDCR: exchange of packets using the low duty cycle mode.
- Stack LLP: communication using the embedded link layer features of the S2-LP STack packet (auto-ack and auto-retransmissions).
- CSMA: exchange of packets between two nodes when an interferer (implemented as a third node set in continuous transmission) disturbs the communication.
- Chat: exchange of strings between two nodes. For this example, the role is symmetric, so it is not necessary to have an A and a B program.

The examples are provided in source format and as an IAR (required toolchain is IAR Embedded Workbench for ARM (EWARM) toolchain (V7.40.3 or higher) and an MDK-ARM Keil project (required toolchain is V5.17 or higher).

5.2.1 IAR project

The workspace file is called S2LPLibrary_Examples.eww and is placed in the directory Projects/Projects_Cube/S2-LP_DK/S2LPLibrary_Examples in the EWARM folder under STM32 or BlueNRG folder.

Each program is an IAR configuration and can be compiled and flashed on the motherboard using the embedded ST-LINK for STM32L053R8/STM32L152RE or the external one for BlueNRG-1/BlueNRG-2 boards.

To use the project with IAR Embedded Workbench for ARM:

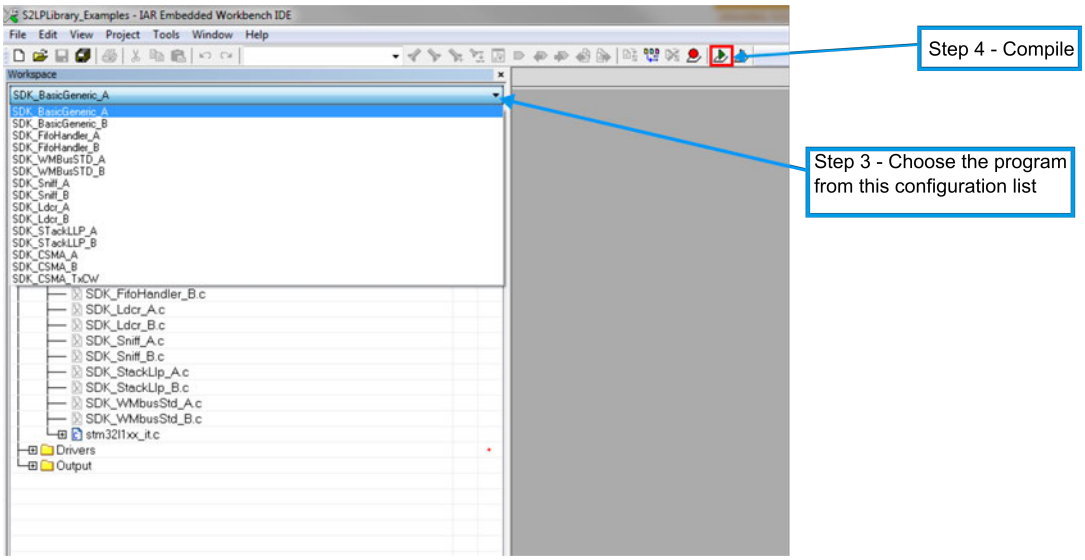
Step 1. Open the Embedded Workbench for ARM and select [**File**]>[**Open**]>[**Workspace**] menu.

Step 2. Open the IAR project .../ Projects/Projects_Cube/S2-LP_DK/S2LPLibrary_Examples and select the EWARM/S2LPLibrary_Examples.eww file under STM32 or BlueNRG folder

Step 3. Select the desired configuration to build

Step 4. Select the download and debug button to recompile and link the entire application, download the related binary image and go in debug mode.

Figure 29. IAR project



5.2.2 MDK-ARM KEIL project

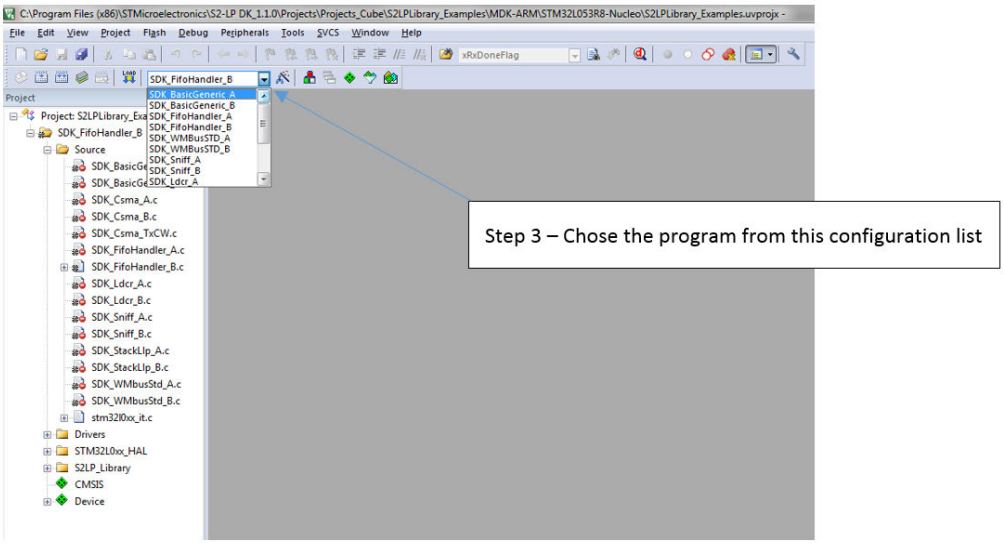
To use the project with KEIL uVision 5 for ARM:

Step 1. Open the KEIL uVision 5 for ARM and select [**Project**]>[**Open Project**] menu

Step 2. Open the KEIL project .../Projects/Projects_Cube/S2-LP_DK/S2LPLibrary_Examples and select the MDK-ARM/S2LPLibrary_Examples.uvprojx file under STM32 or BlueNRG folder

Step 3. Select the desired configuration to build

Figure 30. Keil project



Step 4. Select [**Project**]>[**Rebuild all target files**] to recompile and link the entire application.

Step 5. Select [**Project**]>[**Download**] to download the related binary image.

6 Release notes

The S2-LP DK SW package release notes are contained in Documents/S2LP_DK_release_notes/Release_Notes.html (html format).

Open the file Documents/index.html for a global documentation index.

7 License

The S2-LP DK software package license file is accessible through the Documents/index.html file.

Revision history

Table 14. **Document revision history**

Date	Version	Changes
19-Dec-2016	1	Initial release.
01-Feb-2017	2	Updated Section 1.2: "STSW-S2LP-DK development kit setup", Section 2.1: "STEVAL-FKI868V1 and STEVAL-FKI433V1 evaluation boards", Section 2.2: "STEVAL-FKI915V1 evaluation board", Figure 2: "STEVAL-FKI915V1 evaluation board features", Section 2.3.4: "STM32L microcontroller", Section 3.1: "Installation", Section 3.9.2.1: "Firmware upgrade", Section 4: "Firmware examples" Added Section 4.1: "IAR project" and Section 4.2: "MDK-ARM Keil project"
26-Jul-2017	3	Updated Figure 6: "S2-LP GUI main window", Section 3.2.1: Connection panel", Figure 10: "Basic packet setting panel", Section, 3.3.2: "Packet setting: WMBUS", Section 5: "Firmware examples" and Section 3.4: "Transmission test". Added Section 4: "S2-LP Consumption Tool", Section 4.1: "Wakeup mode", Section 4.2: "Consumption mode", Section 4.3: "TX" and Section 4.4: "Current profile graph".
23-May-2018	4	Updated frequency bands and supported HW. Added new device: STEVAL-FKI512V1.
03-Jul-2019	5	Updated Introduction, Section 1.1 System requirements, Section 2.1 STEVAL-FKI868V2, STEVALFKI433V2 and STEVAL-FKI512V1 evaluation boards, Section 2.1.2 STEVAL-FKI868V2, STEVALFKI433V2 and STEVAL-FKI512V1 power, Section 2.2 STEVAL-FKI915V1 evaluation board, Section 2.2.2 STEVAL-FKI915V1 power, Section 3.1 Installation, Section 3.2.2 Radio setting panel, Section 3.9.2.1 Firmware upgrade, Section 5.2.1 IAR project and Section 5.2.2 MDK-ARM KEIL project. Added Section 2.5 STEVAL-IDB007V2 and STEVAL-IDB008V2 evaluation boards, Section 2.6 STEVAL-FKI001V1 evaluation board, Section 5.1 Command line interface (CLI), Section 5.1.1 CLI commands, modulation types and S2-LP modes, Section 5.1.2 IAR project and Section 5.1.3 MDK-ARM KEIL project.
09-Mar-2020	6	Updated Introduction, Section 1.1 System requirements, Figure 6. S2-LP GUI main window and Section 3.2.2 Radio setting panel. Added Section 2.4 X-NUCLEO-S2915A1.

Contents

List of tables

List of figures