UM1657
User manual

SPIRIT1 development kit

Introduction

The SPIRIT1 development kit is a complete software package to support SPIRIT1 RF evaluation and development.

The kit includes a SPIRIT1 development kit graphical user interface (SPIRIT1 DK - GUI) which allows checking the SPIRIT1 main performance and easily measure parameters such as sensitivity, output power and main features of the SPIRIT1. It also contains SPIRIT1 firmware libraries for STM32L and STM8L to allow development of SPIRIT1 applications.

In addition, it contains a Wireless M-BUS library with documentation and example applications to allow development of Wireless M-BUS application based on the SPIRIT1.

Contents

List of tables

List of figures

1 Hardware description

The software SPIRIT1 DK - GUI supports all the hardware available as demo kit. The hardware includes STEVAL-IKR002Vx (with x = 1, 2, 3, 4, 5), STEVAL-IKR001V7D, STEVAL-IKR002V7D, STEVAL-IKR001V8D, STEVAL-IKR002V4B, STEVAL-IKR001Vx (with x = 1, 2, 3, 4, 5) and STEVAL-IDS001Vx (with x = 2, 3, 4, 5).

1.1 STEVAL-IKR002Vx

The SPIRIT1 DK is made up of two units connected to one PC or two PCs with USB cables.

Each unit consists of an antenna for the selected band and two PCBs:

- RF motherboard
- RF module

1.1.1 STEVAL-IKR002Vx (RF motherboard)

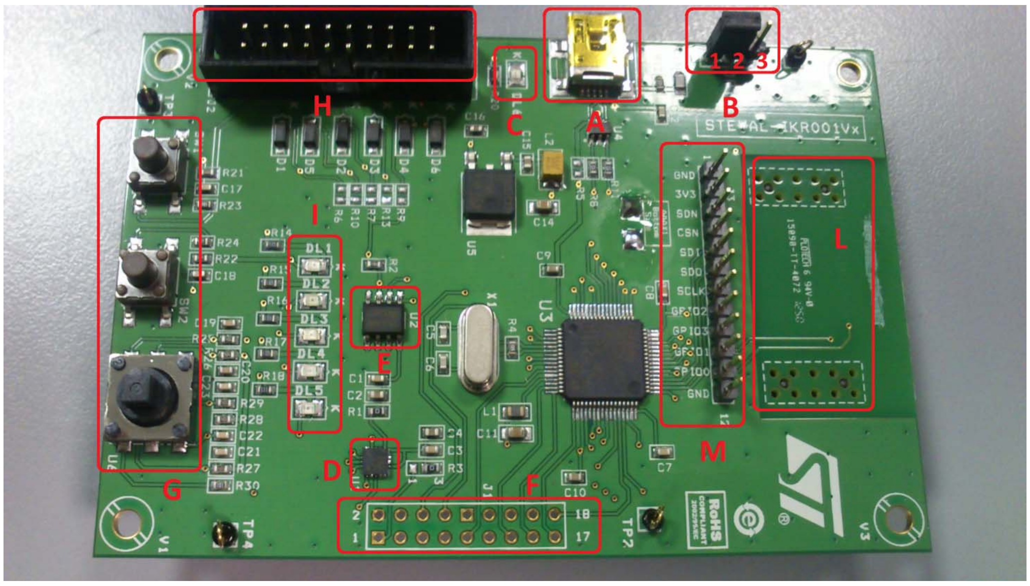
The RF motherboard has an STM32L microcontroller used for driving the SPIRIT1 transceiver and to communicate to a PC via USB.

A connector on the motherboard (*Figure 1*) allows accessing the JTAG interface for programming and debugging. The board can be powered through a mini-USB connector that can also be used for I/O interaction with a USB Host. The board also has a button, a joystick and RESET button for user interaction. Temperature sensor and accelerometer are included in the board. The RF module can be easily connected through a dedicated interface.

This is the list of some of the features that are available on the boards:

- STM32L151RBT6 64-pin microcontroller
- Mini USB connector for power supply and I/O
- JTAG connector
- RF daughterboard interface
- One RESET button and one USER button
- One LIS3DH accelerometer
- One STLM75 temperature sensor
- One joystick
- 5 LEDs
- One PWR LED
- One battery holder for 2 AAA batteries
- One row of test points on the interface with the RF daughterboard

Figure 1. STEVAL-IKR002Vx (RF motherboard)



1.1.2 Microcontroller and connections

The board has an STM32L151RB microcontroller. It is an ultralow power microcontroller with 128 KB of Flash memory, 16 KB of RAM, 32-bit core ARM cortex-M3, 4 KB of data EEPROM, RTC, LCD, timers, USART, I²C, SPI, ADC, DAC and comparators.

The microcontroller is connected to different components like buttons, LEDs and connectors for external circuitry. The following table shows which functionality is available on each microcontroller pin.

Table 1. Motherboard MCU pin description versus function

Pin name	Pin n°	Board function							
		LEDs	**DB connector**	**Buttons / Joystick**	**Accelerometer**	**Temp. sensor**	**USB**	**JTAG**	**Ext. conn.**
VLCD	1								
PC13	2		DB_SDN_RST(1)						
PC14	3								3
PC15	4								5
OSC_IN	5								
OSC_OUT	6								
NRST	7			RESET					7
PC0	8	LED1							
PC1	9	LED2							
PC2	10		DB_PIN3						

Table 1. Motherboard MCU pin description versus function (continued)

Pin name	Pin n°	Board function							
		LEDs	DB connector	Buttons / Joystick	Accelerometer	Temperature sensor	USB	JTAG	Ext. conn.
PC3	11								9
VSSA	12								
VDDA	13								
PA0	14								11
PA1	15								13
PA2	16								15
PA3	17								17
VSS_4	18								
VDD_4	19								
PA4	20				SPI1_NSS				
PA5	21				SPI1_SCK				
PA6	22				SPI1_MISO				
PA7	23				SPI1_MOSI				
PC4	24	LED4							
PC5	25	LED5							
PB0	26			JOY_DOWN					
PB1	27			JOY_RIGHT					
PB2	28								18
PB10	29				INT1				
PB11	30				INT2				
VSS_1	31								
VDD_1	32								
PB12	33		DB_CSN*						
PB13	34		DB_SCLK*						
PB14	35		DB_SDO*						
PB15	36		DB_SDI*						
PC6	37			PUSH_BTN					
PC7	38		DB_IO0*						
PC8	39		DB_IO1*						
PC9	40		DB_IO2*						

Table 1. Motherboard MCU pin description versus function (continued)

Pin name	Pin n°	Board function							
		LEDs	DB connector	Buttons/ Joystick	Accelerometer	Temperat ure sensor	USB	JTAG	Ext. conn.
PA8	41			JOY_LEFT					
PA9	42			JOY_CEN TER					
PA10	43			JOY_UP					
PA11	44						USB_D M		
PA12	45						USB_D P		
PA13	46							JTMS	16
VSS_2	47								
VDD_2	48								
PA14	49							JTCK	14
PA15	50							JTDI	12
PC10	51		DB_IO3_IRQ*						
PC11	52		DB_PIN1						
PC12	53		DB_PIN2						
PD2	54	LED3							
PB3	55							JTDO	10
PB4	56							JNTRS T	8
PB5	57					TSEN_IN T			
PB6	58					I2C1_SCL			
PB7	59					I2C1_SDA			
BOOT0	60								
PB8	61								4
PB9	62								6
VSS_3	63								
VDD_3	64								

1. These lines are also available on the test point row.

1.1.3 Power

The board can be powered either by the mini USB connector CN1 (A in *Figure 1*) or 2 AAA batteries. To power the board through USB bus, jumper JP1 must be in position 1-2, as in *Figure 1* (B). To power the board by batteries, 2 AAA batteries must be present in the battery holder in the rear of the board and the jumper JP1 must be set in position 2-3.

When the board is powered, the green LED DL6 is ON, *Figure 1* (C).

If needed, the board can be powered by an external DC power supply. Connect the positive output of the power supply to the central pin of JP1 (pin 2) and the ground to one of the four test point connectors on the motherboard.

1.1.4 Sensors

Two sensors are available on the motherboard:

- LIS3DH, an ultra-low power high performance three axes linear accelerometer (D in *Figure 1*). The sensor is connected to the STM32L through the SPI interface. Two lines for interrupts are also connected.
- STLM75, a high precision digital CMOS temperature sensor, with I²C interface (E in *Figure 1*). The pin for the alarm function is connected to one of the STM32L GPIOs.

1.1.5 Extension connector

There is the possibility to solder a connector on the motherboard to extend its functionality (F in *Figure 1*). 16 pins of the microcontroller are connected to this expansion slot (*Table 1*).

1.1.6 Daughterboard test point

In “M” (*Figure 1*), a row of test point is available to the user for debugging and testing. The list of signals available is:

1. GND
2. VDD
3. GPIO SDN (the pin to drive in SHUTDOWN the SPIRIT1)
4. SPI CSn
5. SPI MISO
6. SPI MOSI
7. SPI SCKL
8. SPIRIT1 GPIO2
9. SPIRIT1 GPIO3
10. SPIRIT1 GPIO1
11. SPIRIT1 GPIO0
12. GND

1.1.7 Push buttons and joystick

For user interaction, the board has two buttons and a joystick. One is to reset the microcontroller; the other one is available for the application. There is also a digital joystick with four possible positions (left, right, up, down) (G in *Figure 1*).

1.1.8 JTAG connector

A JTAG connector on the board (H in *Figure 1*) allows programming and debugging of the STM32L microcontroller on board, using an in-circuit debugger and programmer like the ST-LINK/V2.

1.1.9 LEDs

Five LEDs are available (I in *Figure 1*).

– DL1: green.
– DL2: orange.
– DL3: red.
– DL4: blue
– DL5: yellow.

1.2 STEVAL-IKR002Vx (RF module)

The RF module includes 5 different possible BOM lists, on the same layout PCB. Each one optimized for different RF band as follows:

- 169 MHz
- 315 MHz
- 433 MHz
- 868 MHz
- 915 MHz

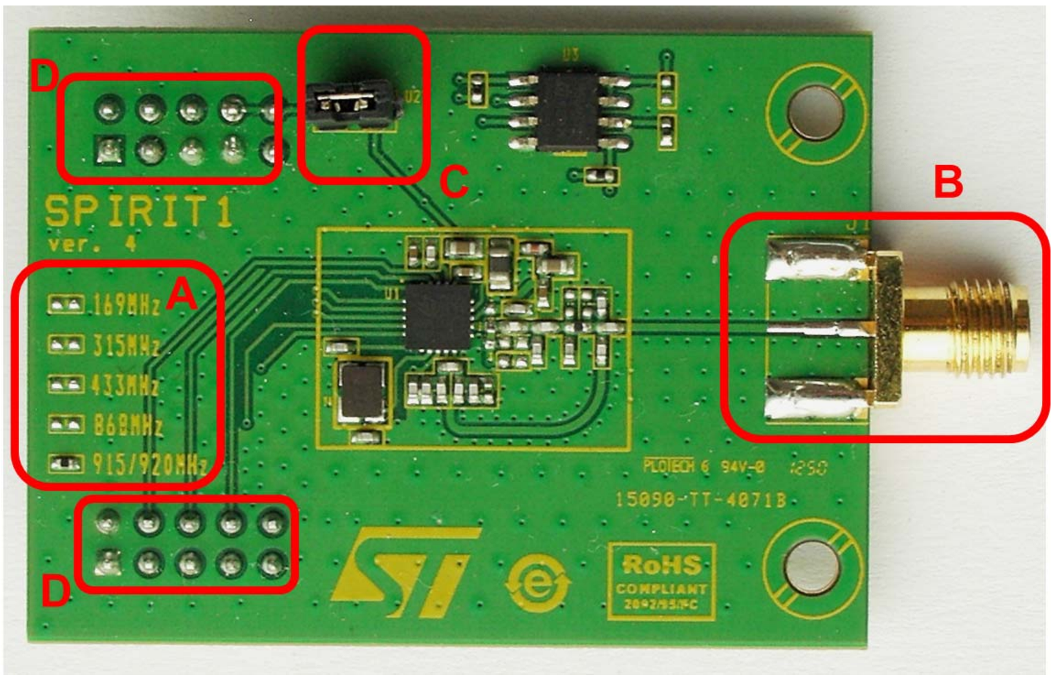
The band indication is reported with a dummy resistor on the board, in A (*Figure 2*).

An SMA connector on the RF module, in B (*Figure 2*), allows connection with RF instruments as spectrum analyzer and signal generator to the SPIRIT1 by RF cable or also connect an antenna as the one included in the demo kit. The board is powered through the RF daughterboard connector, in D *Figure 2*, by the RF motherboard and communicates through this connector by SPI and some GPIOs with the microcontroller.

The VCC_RF pin of the RF daughterboard connector is linked to the Vbat of the SPIRIT1 through a jumper that can be also removed to measure the current consumption, in C *Figure 2*

The RF module includes a memory EEPROM in which some information on the RF module at manufactory time are stored. The information is stored in the first pages of the EEPROM. The memory is not intended to be changed by user.

Figure 2. STEVAL-IKR002Vx (RF module)



1.2.1 Boost mode

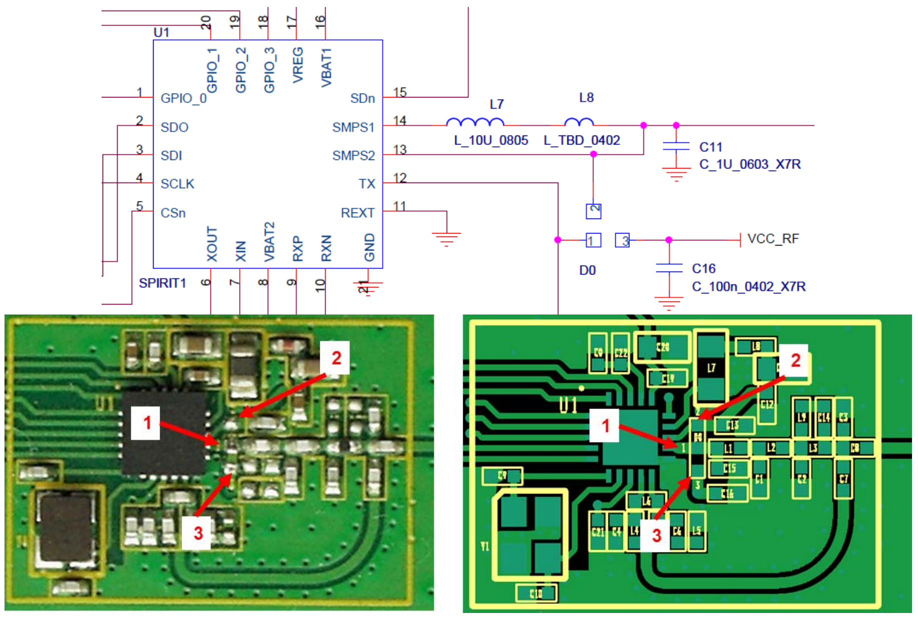
The SPIRIT1 can be configured to increase the output power in transmission mode.

In the default configuration, the transmitter power amplifier (PA) output is biased by the 1.4 V SMPS voltage output through the L0 external inductor (position D0 in the schematic, *Figure 3*). This limits the maximum output power at about +11 dBm, measured at the 50 Ω connector via the reference design.

Biasing the PA output through the inductor L0 directly connected to the battery, instead of the SMPS output allows the maximum output power delivered at the 50 Ω connector (or at the antenna) to be increased. The maximum output power changes with the voltage level applied at the PA output.

To switch to the boost mode the inductor L0 must be removed from the position 1-2 D0 in the schematic and soldered at the position 1-3 D0, then the voltage supply VCC_RF must be provided.

Figure 3. STEVAL-IKR002Vx (RF module) boost mode configuration



For more information, refer to the application note AN4198 - SPIRIT1: increasing the output power, available on the SPIRIT1 website or in the document folder of this release.

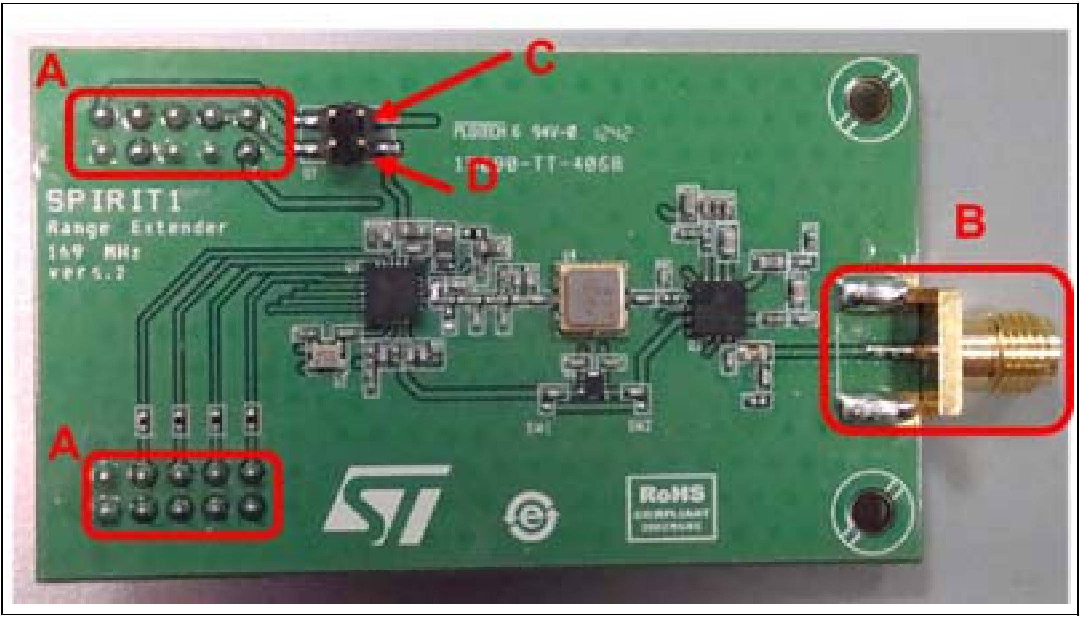
1.3 STEVAL-IKR001V7D (with SPIRIT1 RF module with external power amplifier @169 MHz)

The STEVAL-IKR001V7D (refer to *Figure 4*) includes RF motherboard and SPIRIT1 RF module with external power amplifier tuned for 169 MHz band. An SMA connector on the RF module,(in B) allows connection with RF instruments as spectrum analyzer and signal generator to the SPIRIT1 by RF cable or also connect an antenna as the one included in the demo kit. The board is powered through the RF daughterboard connector by the RF motherboard and communicates through this connector by SPI and some GPIOs with the microcontroller.

A SAW filter is recommended to attenuate the spurious and harmonics emissions above the carrier frequency, which would otherwise violate spurious emissions limits under ETSI 300 220.The VCC_RF pin of the RF daughterboard connector is linked to the Vbat of the SPIRIT1 through the jumper D that can be also removed to measure the current consumption, while the jumper C is used to provide voltage to the power amplifier (3.3 V).

The RF module includes a memory EEPROM in which some information on the RF module at manufactory time are stored. The information is stored in the first pages of the EEPROM. The memory is not intended to be changed by user.

Figure 4. STEVAL-IKR001V7D (RF module)



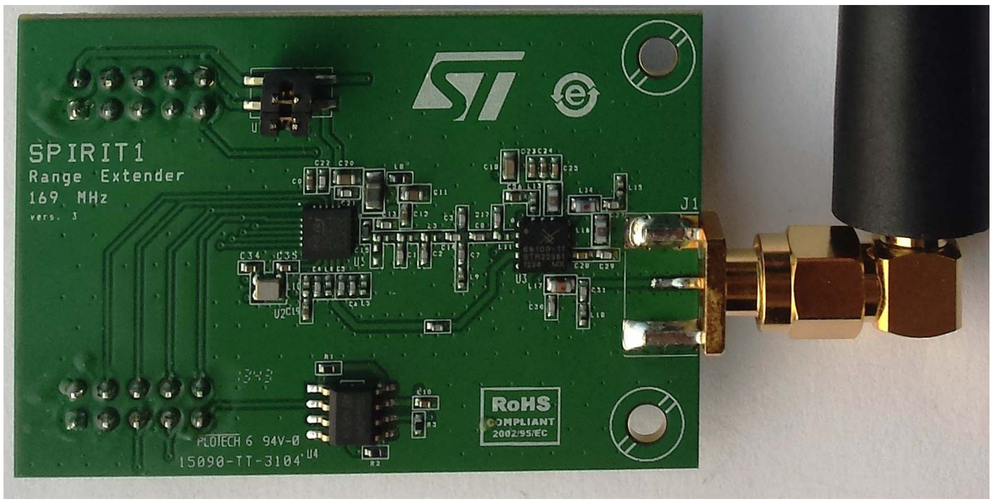
1.4 STEVAL-IKR002V7D (with SPIRIT1 RF module with external power amplifier @169MHz)

The STEVAL-IKR002V7D, illustrated in *Figure 4*, includes RF motherboard and SPIRIT1 RF module with external power amplifier tuned for 169 MHz band. An SMA connector on the RF module (B) allows connection with RF instruments as spectrum analyzer and signal generator to the SPIRIT1 by RF cable or also connect an antenna as the one includedin the demo kit. The board is powered through the RF daughterboard connector (A) by the RF motherboard and communicates through this connector by SPI and some GPIOs with the microcontroller.

The Vcc_RF pin of the RF daughterboard connector is linked to the Vbat of the SPIRIT1 through the jumper D that can be also removed to measure the current consumption, while the jumper C is used to provide voltage to the power amplifier (3.3 V).

The RF module includes a memory EEPROM in which some information on the RF module at manufactory time are stored. The information is stored in the first pages of the EEPROM. The memory is not intended to be changed by the user.

Figure 5. STEVAL-IKR002V7D



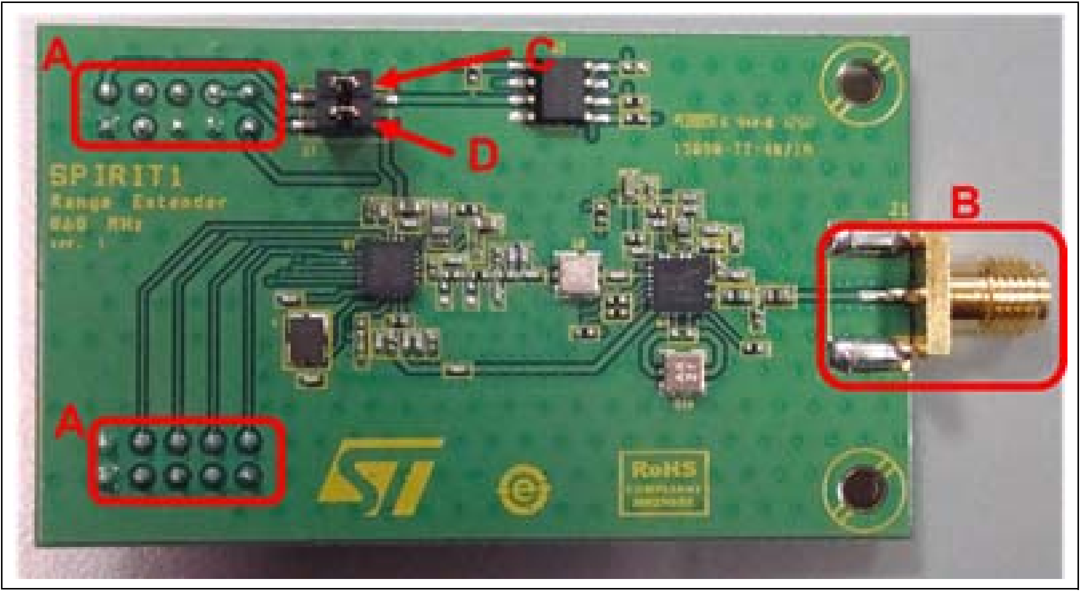
1.5 STEVAL-IKR001V8D (with SPIRIT1 RF module with external power amplifier @868 MHz)

The STEVAL-IKR001V8D (*Figure 6*) includes RF motherboard and SPIRIT1 RF module with external power amplifier tuned for 169 MHz band. An SMA connector on the RF module, in B, allows connection with RF instruments as spectrum analyzer and signal generator to the SPIRIT1 by RF cable or also connect an antenna as the one included in the demo kit. The board is powered through the RF daughterboard connector, in A, by the RF motherboard and communicates through this connector by SPI and some GPIOs with the microcontroller.

A SAW filter is recommended in the TX path to attenuate the spurious emissions above the carrier frequency, which would otherwise violate spurious emissions limits under ETSI 300 220. The same SAW filter is provided in the RX path because, due the proximity of the GSM band, the receiver can be saturated in an application environment.The VCC_RF pin of the RF daughterboard connector is linked to the Vbat of the SPIRIT1 through the jumper D in that can be also removed to measure the current consumption, while the jumper C is used to provide voltage to the power amplifier.

The RF module includes a memory EEPROM in which some information on the RF module at manufactory time are stored. The information is stored in the first pages of the EEPROM. The memory is not intended to be changed by user.

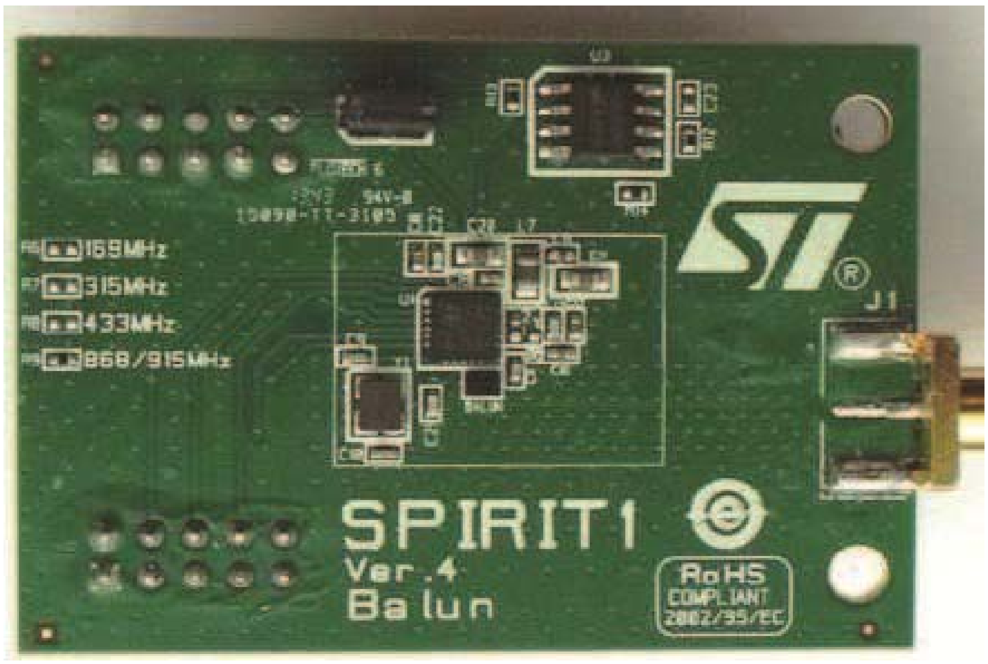
Figure 6. STEVAL-IKR001V8D (RF module)



1.6 STEVAL-IKR002V4B (with SPIRIT1 RF module and integrated balun)

The same boards as STEVAL-IKR002Vx is available with an integrated version of the balun instead of an SMD composed one.

Figure 7. STEVAL-IKR002V4B (RF module)



1.7 STEVAL-IDS001Vx (SPIRIT1 USB dongle)

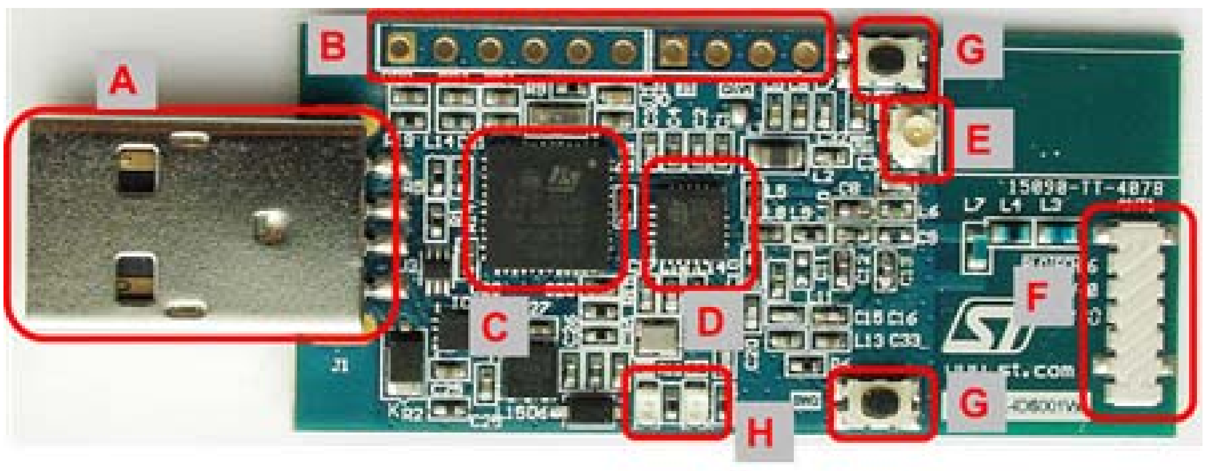
The STEVAL-IDS001Vx (*Figure 8*) is a single board that includes the SPIRIT1 radio transceiver and STM32L microcontroller.

A strip line can be soldered, in B, allowing the access to the SWD interface for programming and debugging and to the 4 GPIOs of the SPIRIT1. The board can be powered through the USB connector, in A, that can also be used for I/O interaction with a USB Host. The board has also two user buttons for user interaction, in G.

This is the list of some of the features that are available on the module:

- SPIRIT1 radio transceiver, in D.
- STM32L151CBU6 48-pin microcontroller, in C.
- USB connector for power supply and I/O, in A.
- One row of pins with SWD interface and SPIRIT1's GPIOs, in B.
- Chip antenna tuned for the entire bands supported (315 MHz, 433 MHz, 868 MHz and 915 MHz), in F.
- Two user buttons, in G.
- 2 LEDs, in H.

Figure 8. STEVAL-IDS001Vx



1.7.1 Microcontroller and connections

The board has an STM32L151CBU6 microcontroller. It is an ultralow power microcontroller with 128 KB of Flash memory, 16 KB of RAM, 32-bit core ARM cortex-M3, 4 KB of data EEPROM, RTC, timers, USART, I²C, SPI, ADC, DAC and comparators.

The microcontroller is connected to different components like buttons, LEDs and connectors for external circuitry. The following *Table 2* shows which functionality is available on each microcontroller pin.

Table 2. Dongle MCU pin description versus function

Pin name	Pin n°	Board function					
		LEDs	SPIRIT1	Buttons	USB	SWD	Ext. conn.
VLCD	1		VBAT				
PC13	2		SDN_SPIRIT1				
PC14	3						
PC15	4						
OSC_IN	5						
OSC_OUT	6						
NRST	7						2
VSS_A	8						
VDD_A	9						
PA0	10						
PA1	11			USER_BUTTON_2			
PA2	12						
PA3	13						
PA4	14						
PA5	15						
PA6	16						
PA7	17						
PB0	18	USER_LED_2					
PB1	19	USER_LED_1					
PB2	20			USER_BUTTON_1			
PB10	21		GPIO3_SPIRIT1				9
PB11	22						
VSS1	23						
VDD1	24						
PB12	25		SPI1_CS				
PB13	26		SPI1_CLK				
PB14	27		SPI1_MISO				
PB15	28		SPI1_MOSI				
PA8	29						
PA9	30						
PA10	31						
PA11	32				USB_DM		

Table 2. Dongle MCU pin description versus function (continued)

Pin name	Pin n°	Board function					
		LEDs	SPIRIT1	Buttons	USB	SWD	Ext. conn.
PA12	33				USB_DP		
PA13	34					SWDIO	3
VSS2	35						
VDD2	36						
PA14	37					SWCLK	5
PA15	38						
PB3	39					SWO/TRACE	4
PB4	40						
PB5	41						
PB6	42						
PB7	43		GPIO0_SPIRIT1				6
BOOT0	44						
PB8	45		GPIO1_SPIRIT1				7
PB9	46		GPIO2_SPIRIT1				8
VSS_3	47						
VDD_4	48						

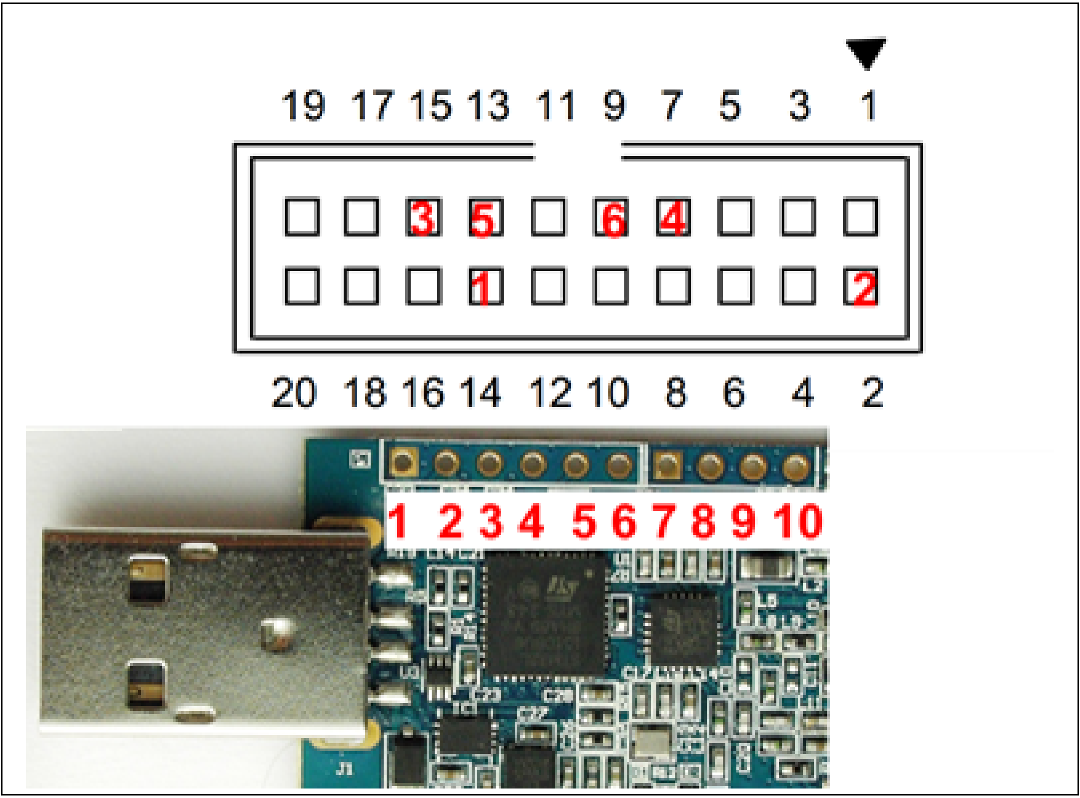
1.7.2 Extension connector

There is the possibility to solder a strip line connector on the motherboard to access to the SWD interface for programming and debugging, and to access to the 4 GPIOs of the SPIRIT1, in B *Figure 8*.

1.7.3 SWD interface

The SWD interface is available through the pins in B in *Figure 8*. The SWD interface allows programming and debugging of the STM32L microcontroller on board, using an in-circuit debugger and programmer like the ST-LINK/V2. In *Figure 9* the connection scheme to how connect the ST-LINK/V2 with the board pins is showed.

Figure 9. SWD connection scheme with ST-LINK/V2



The signals available on the STEVAL-IDS001Vx are:

1. GND.
2. VDD.
3. nRESET.
4. SWDIO.
5. SWO/TRACE.
6. SWCLK.
7. SPIRIT1 GPIO0.
8. SPIRIT1 GPIO1.
9. SPIRIT1 GPIO2.
10. SPIRIT1 GPIO3.

The connection with the ST-LINK/V2 interface is illustrated in *Figure 9*:

Table 3. SWD connection

Signal name	STEVAL-IDS001Vx pin number	ST-LINK/V2 pin number
GND	1	14
VDD	2	2
nRESET	3	15
SWDIO	4	7

Table 3. SWD connection (continued)

Signal name	STEVAL-IDS001Vx pin number	ST-LINK/V2 pin number
SWO/TRACE	5	13
SWCLK	6	9

1.7.4 Band

The STEVAL-IDS001Vx has 4 different possible BOM lists, each one optimized for different RF band as follow:

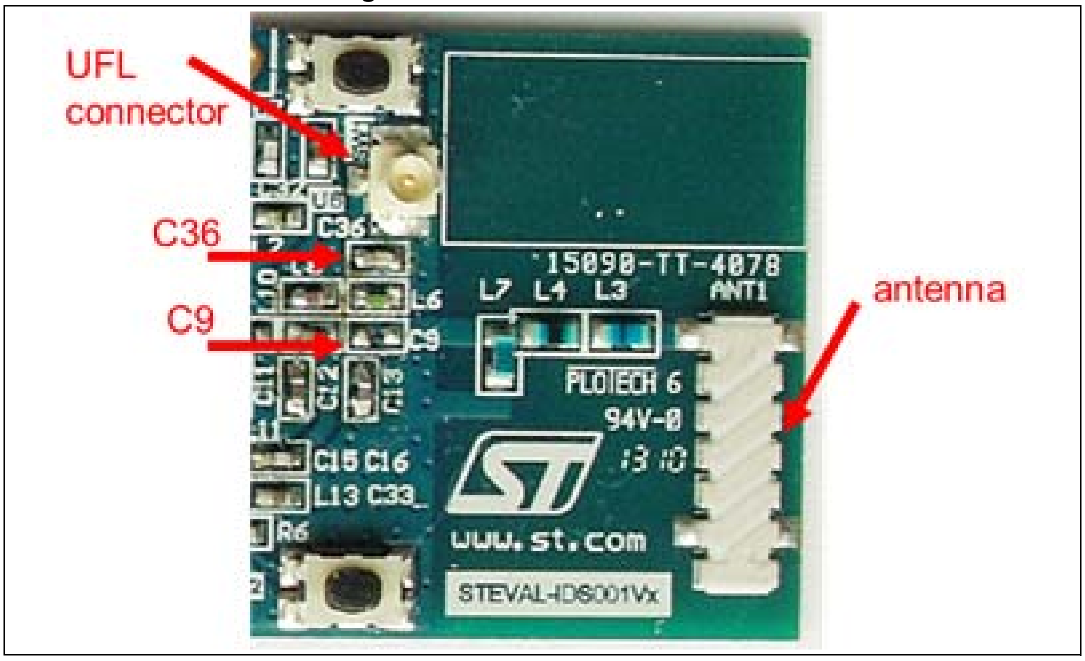
- 315 MHz
- 433 MHz
- 868 MHz
- 915 MHz

The chip antenna used supports all these four bands of frequency.

1.7.5 RF connector

The STEVAL-IDS001Vx provides two different RF connections: antenna (chip antenna tuned for all the supported bands, default configuration) and UFL connector. Although the default configuration allows communication on air, can be useful switch to the UFL connector to connect the STEVAL-IDS001Vx to RF equipment as spectrum analyzer or RF signal generator. To switch from antenna to UFL connector the C9 capacitor must be removed and the C36 capacitor must be soldered. To restore the default configuration and use the antenna the C36 capacitor must be removed, and the C9 capacitor must be soldered. In *Figure 10*, the two pads for C9 and C36 are showed together with the chip antenna and UFL connector. According to the particular BOM, for each band, the value of C36 equals to C9.

Figure 10. RF connector scheme



1.7.6 Push buttons

For user interaction, the board has two buttons both available to the application, G in *Figure 8*.

1.7.7 LEDs

Two LEDs are available (H in *Figure 8*).

– D1: yellow.

– D2: green.

1.8 STEVAL-IKR001Vx

The SPIRIT1 DK is made up of two units connected to one PC or two PCs with USB cables.

Each unit is composed of an antenna for the selected band and two PCBs:

- RF motherboard
- RF module

1.8.1 STEVAL-IKR001Vx (RF motherboard)

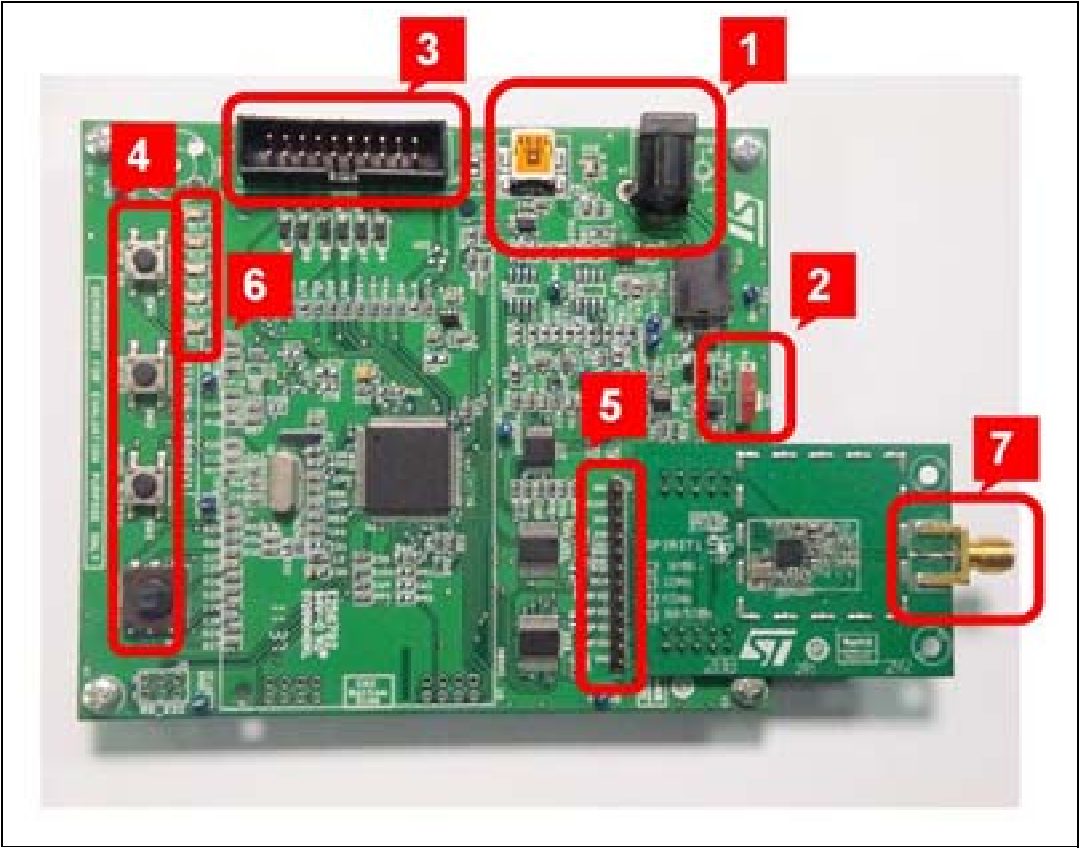
The previous version of the motherboard is supported.

The supply voltage is provided or by USB cable or external power supply. The switch S1 turns on or off the SPIRIT1 DK - MB.

Figure 11 highlights some features of the board as follow:

1. The power supply sources: USB connector and jack connector for external power supply. USB connector is also used for I/O with the microcontroller with a virtual COM port exposed to the PC.
2. Switch to turn ON or OFF the board.
3. JTAG/SWD connector.
4. Three buttons and a joystick:
   a) SCM_PS, not used
   b) RESET, for resetting the MB board
   c) Push_Button, used to enter in DFU mode
   d) Joystick, not used.
5. Test point (TP), to probe the SPI signals, the 4 GPIOs of the SPIRIT1, the shutdown of the SPIRIT1 (SDN), the supply voltage of the SPIRIT1 (VCCRF). The list of signals available is:
   a) GND
   b) VCC_RF
   c) GPIO SDN (the pin to drive in SHUTDOWN the SPIRIT1)
   d) SPI CSn
   e) SPI MISO
   f) SPI MOSI
   g) SPI SCKL
   h) SPIRIT1 GPIO2
   i) SPIRIT1 GPIO3
   j) SPIRIT1 GPIO1
   k) SPIRIT1 GPIO0
   l) GND
6. Five LEDs
7. SMA connector

Figure 11. STEVAL-IKR001Vx



1.8.2 STEVAL-IKR001Vx (RF module)

The RF module includes 5 different possible BOM lists, on the same layout PCB. Each one optimized for different RF band as follow:

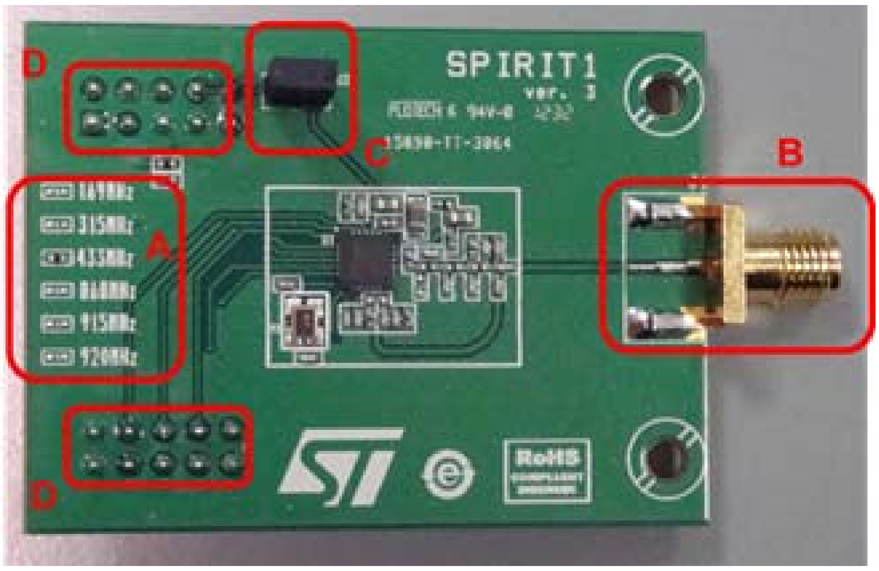
- 169 MHz
- 315 MHz
- 433 MHz
- 868 MHz
- 915 MHz

The band indication is reported with a dummy resistor on the board, in A *Figure 12*.

A SMA connector on the RF module, in B *Figure 12*, allows connection with RF instruments as spectrum analyzer and signal generator to the SPIRIT1 by RF cable or also connect an antenna as the one included in the demo kit. The board is powered through the RF daughterboard connector, in D *Figure 12*, by the RF motherboard and communicates through this connector by SPI and some GPIOs with the microcontroller.

The VCC_RF pin of the RF daughterboard connector is linked to the Vbat of the SPIRIT1 through a jumper that can be also removed to measure the current consumption, in C *Figure 12*.

Figure 12. STEVAL-IKR001Vx (RF module)

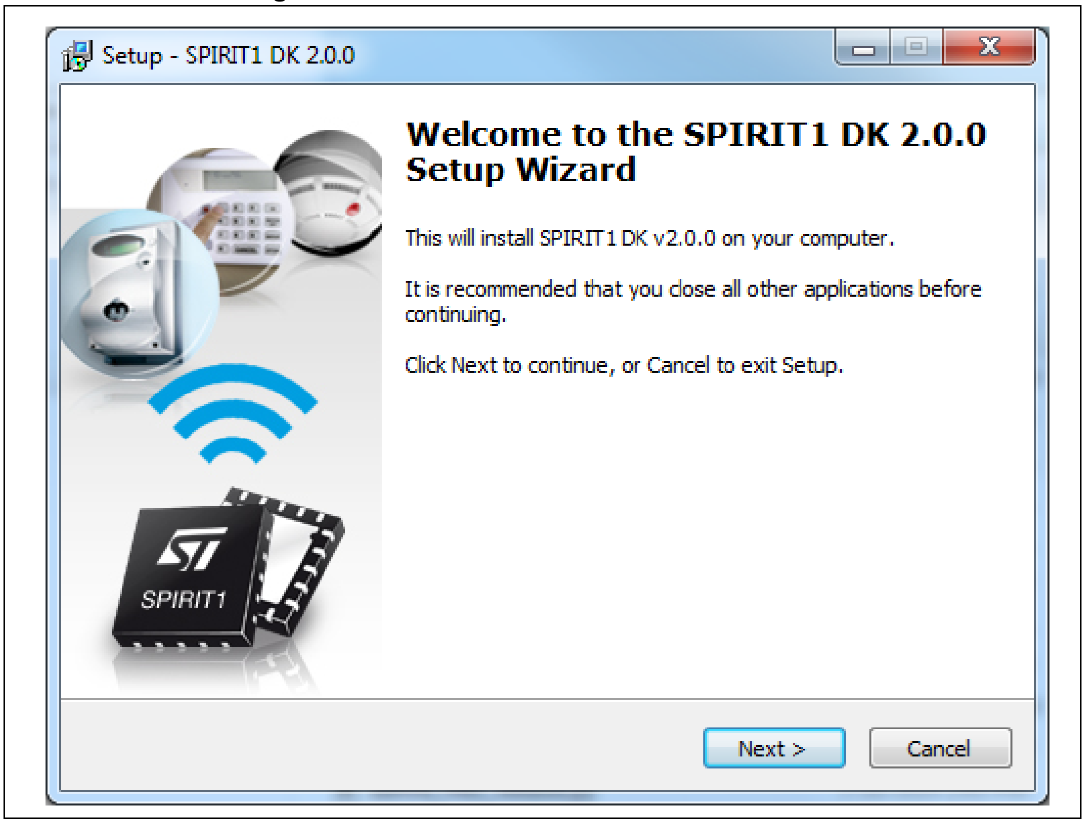


2 Software installation

The SPIRIT1 DK comes as a self-installer file which takes care of installing all the necessary files on the user PC.

To begin the installation, the user must double-click on the installer file and follow the instructions.

Figure 13. SPIRIT DK - GUI installation window



Setup - SPIRIT1 DK 2.0.0

Welcome to the SPIRIT1 DK 2.0.0 Setup Wizard

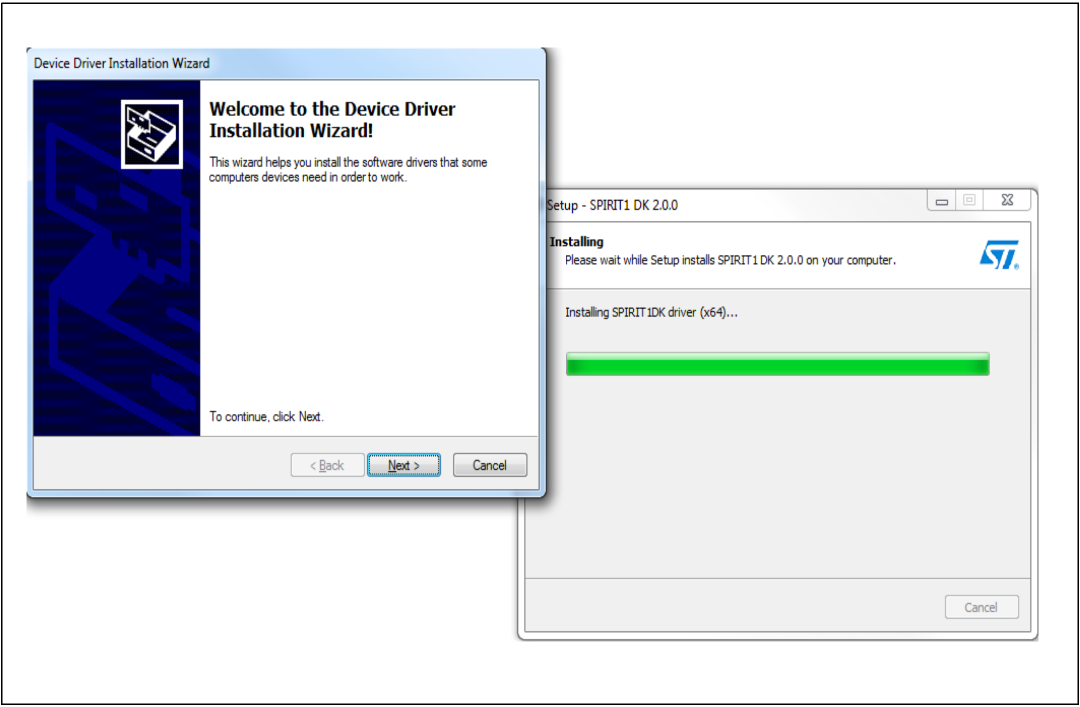
This will install SPIRIT1 DK v2.0.0 on your computer.

It is recommended that you close all other applications before continuing.

Click Next to continue, or Cancel to exit Setup.

Once SPIRIT1 DK - GUI is installed, automatically the driver installation to use the SPIRIT DK - MB starts

Figure 14. SPIRIT DK - driver installation window



Device Driver Installation Wizard

Welcome to the Device Driver Installation Wizard!

This wizard helps you install the software drivers that some computers devices need in order to work.

To continue, click Next.

Setup - SPIRIT1 DK 2.0.0

Installing

Please wait while Setup installs SPIRIT1 DK 2.0.0 on your computer.

Installing SPIRIT1DK driver (x64)...

The user may install the driver also manually make run the "DPInst_x86.exe" or "dpinst_ia64.exe" in "\SPIRIT1 DK\Driver" folder. At first connection of the SPIRIT DK - MB, the user is required to select the proper driver.

At the end of installation, the user can run the SPIRIT1 DK - GUI.

3 SPIRIT1 DK - GUI description

This section describes how to use the Spirit1 DK-GUI.

3.1 Firmware installation

The boards come preprogrammed with firmware suitable to run Spirit1 DK-GUI, but if you have received a new software release with a new firmware version or if you have programmed other firmware in the board, it is necessary to reprogram the appropriate firmware to operate the GUI. In order to restore the firmware, please follow the instruction in section 4.2.9.2 and load the firmware image: ...\Firmware\Binary\SPIRIT1_CLI.hex.

3.2 Detailed description

The SPIRIT1 DK - GUI can use only one SPIRIT1 DK - MB plus SPIRIT1 RF - DB plugged into it, connected through a USB cable to a PC. So, it is necessary to run one instance of SPIRIT1 DK - GUI for each board connected to the PC. *Figure 15* and *Figure 16* shows two typical connections with one or two.

Figure 15. CONNECTION SETUP 1 - 1 PC WITH 2 SPIRIT DK - GUI

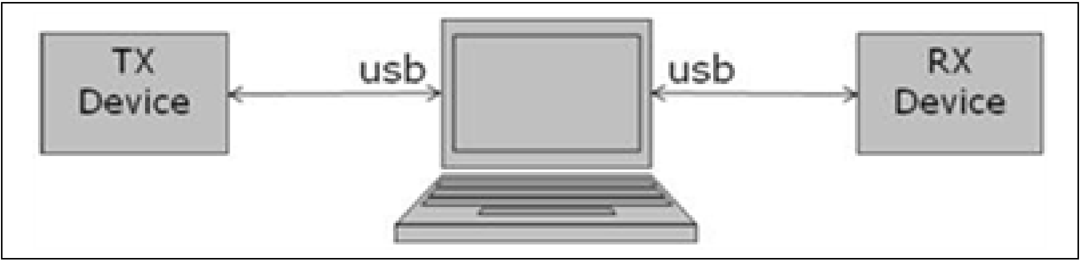
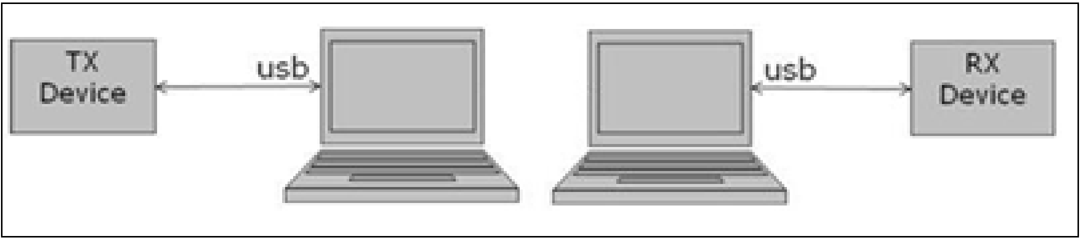


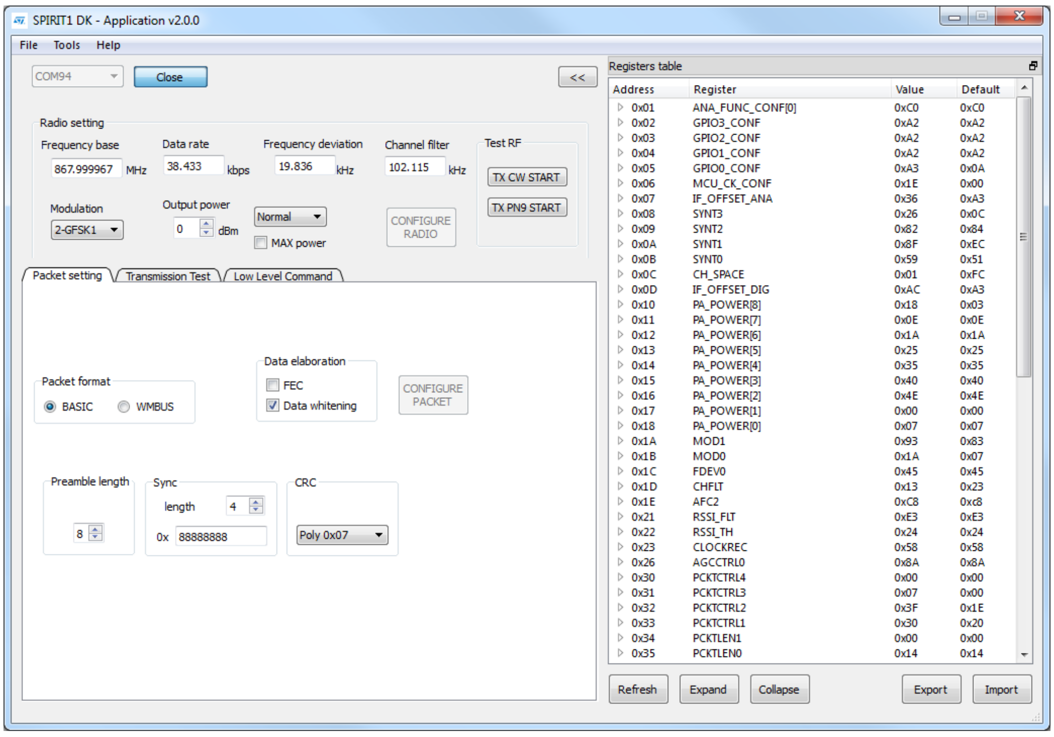
Figure 16. CONNECTION SETUP 2 - 2 PC WITH 1 SPIRIT DK - GUI EACH



During the tests, each SPIRIT1 DK - DB can work indifferently as transmitter (TX) or receiver (RX). The TX device is intended to be used as a transmitter during the communication tests; the RX device is designed to be used as a receiver during the communication tests. The user can configure the SPIRIT1 DK - DB as a TX device or RX device, and dynamically change this selection before running a test.

When the user runs the SPIRIT1 DK.exe file, the SPIRIT1 DK - GUI windows appears as shown below:

Figure 17. Spirit1 DK GUI main window



COM94

Radio setting

Frequency base: 867.999967 MHz
Data rate: 38.433 kbps
Frequency deviation: 19.836 kHz
Channel filter: 102.115 kHz
Test RF
Modulation: 2-GFSK1
Output power: 0 dBm
Normal
MAX power

Packet format: [✓ BASIC] WMBUS
Data elaboration: FEC; [✓ Data whitening]
Preamble length: 8
Sync: length 4; 0x 88888888
CRC: Poly 0x07

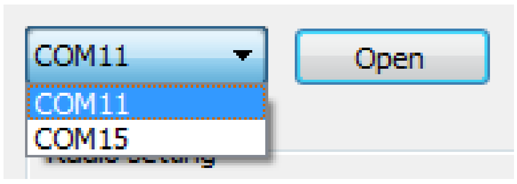
Registers table

Address	Register	Value	Default
0x01	ANA_FUNC_CONF[0]	0xC0	0xC0
0x02	GPIO3_CONF	0xA2	0xA2
0x03	GPIO2_CONF	0xA2	0xA2
0x04	GPIO1_CONF	0xA2	0xA2
0x05	GPIO0_CONF	0xA3	0x0A
0x06	MCU_CK_CONF	0x1E	0x00
0x07	IF_OFFSET_ANA	0x36	0xA3
0x08	SYNT3	0x26	0x0C
0x09	SYNT2	0x82	0x84
0x0A	SYNT1	0x8F	0xEC
0x0B	SYNT0	0x59	0x51
0x0C	CH_SPACE	0x01	0xFC
0x0D	IF_OFFSET_DIG	0xAC	0xA3
0x10	PA_POWER[8]	0x18	0x03
0x11	PA_POWER[7]	0x0E	0x0E
0x12	PA_POWER[6]	0x1A	0x1A
0x13	PA_POWER[5]	0x25	0x25
0x14	PA_POWER[4]	0x35	0x35
0x15	PA_POWER[3]	0x40	0x40
0x16	PA_POWER[2]	0x4E	0x4E
0x17	PA_POWER[1]	0x00	0x00
0x18	PA_POWER[0]	0x07	0x07
0x1A	MOD1	0x93	0x83
0x1B	MOD0	0x1A	0x07
0x1C	FDEV0	0x45	0x45
0x1D	CHFLT	0x13	0x23
0x1E	AFC2	0xC8	0xc8
0x21	RSSI_FLT	0xE3	0xE3
0x22	RSSI_TH	0x24	0x24
0x23	CLOCKREC	0x58	0x58
0x26	AGCCTRL0	0x8A	0x8A
0x30	PCKTCTRL4	0x00	0x00
0x31	PCKTCTRL3	0x07	0x00
0x32	PCKTCTRL2	0x3F	0x1E
0x33	PCKTCTRL1	0x30	0x20
0x34	PCKTLEN1	0x00	0x00
0x35	PCKTLEN0	0x14	0x14

3.2.1 Connection panel

At the top of the main window, the user can select the appropriate COM from a drop down list of the COM port available. Once the correct SPIRIT1 DK - DB COM port is selected, and the "Open" button pressed, and the default configuration of SPIRIT1 DK - DB is loaded and displayed on the SPIRIT1 DK - GUI. Click to open the COM list also makes a refresh of the COM port available.

Figure 18. Serial port selection



3.2.2 Radio setting panel

The "Radio setting" panel is always shown informing the user about:

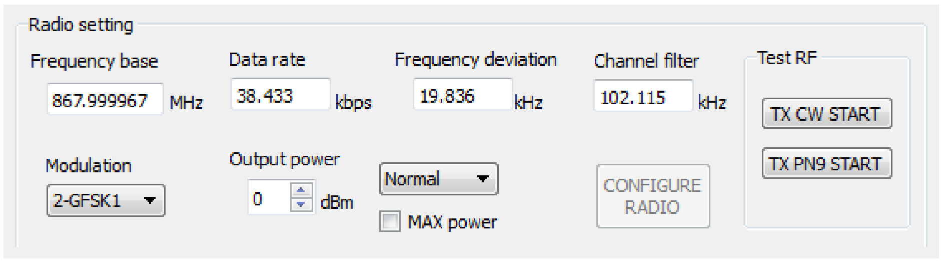
- Frequency base.
- Modulation.
- XTAL frequency.
- Data rate.
- Frequency deviation.
- Channel filter.
- Output power.

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

- Frequency base
  - Low band: [150 - 174] MHz.
  - Middle band I: [300 - 348] MHz.
  - Middle band II: [387 - 470] MHz.
  - High band: [780 - 956] MHz.
- Modulation:
  - 2-FSK.
  - GFSK BT 0.5.
  - GFSK BT 1.
  - ASK.
  - OOK.
- Data rate in the interval: [100 - 500000] bps.
- Frequency deviation in the interval: [793 - 761718] Hz.
- Channel filter in the interval: [1100 - 800100] Hz.
- Output power in the interval: [-30.0 13.0] dBm or [-3.0 +30.0] dBm if PA EXT is checked.
- XTAL frequency is the value of the frequency of the crystal in the SPIRIT1 DK - DB, the value must be in the range: [24 26] MHz and [48 52] MHz. If the AUTO check box is checked, the microcontroller find automatically the XTAL frequency and the text box is only read. Otherwise, if the AUTO check box is not checked, the user must write the correct XTAL frequency value.

By pressing the "Configure radio" button, all the values are sent to the SPIRIT1 DK - DB and then read and displayed for the user.

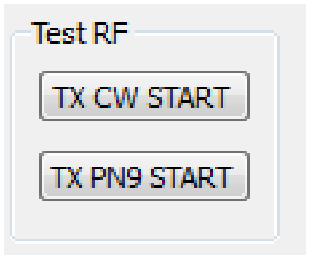
Figure 19. Radio setting panel



3.2.3 RF test mode

Some useful commands are available to put the SPIRIT1 DK - DB in test mode: TX CW and TX PN9.

Figure 20. RF test mode buttons



Both tests require only one device connected to PC.

3.2.4 TX CW test

This test mode can be active using the following steps:

1. Select the desired radio settings and load it by pressing the "Configure radio" button
2. Pressing the "TX CW START" button.

This test mode makes SPIRIT1 DK - DB 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 SPIRIT1 DK - DB stays in TX state until the "TX CW STOP" button is pressed.

When the user wants to change frequency or output power, the running test must be stopped first and then repeat steps 1 and 2 selecting the desired frequency or output power during step 1.

3.2.5 TX PN9 test

This test mode can be activated using the following steps:

1. Select the desired radio setting and then pressed the "Configure radio" button
2. Pressing the "TX PN9 START" button.

This RF test makes SPIRIT1 DK - DB transmits a PN9 data stream modulated according the radio setting.

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

The SPIRIT1 DK - DB stays in state TX until the "TX PN9 STOP" button is pressed.

When the user wants to change the frequency or output power or modulation scheme, the running test must be stopped first and then repeat steps 1 and 2 selecting the desired frequency or output power or modulation scheme during step 1.

3.2.6 Packet setting

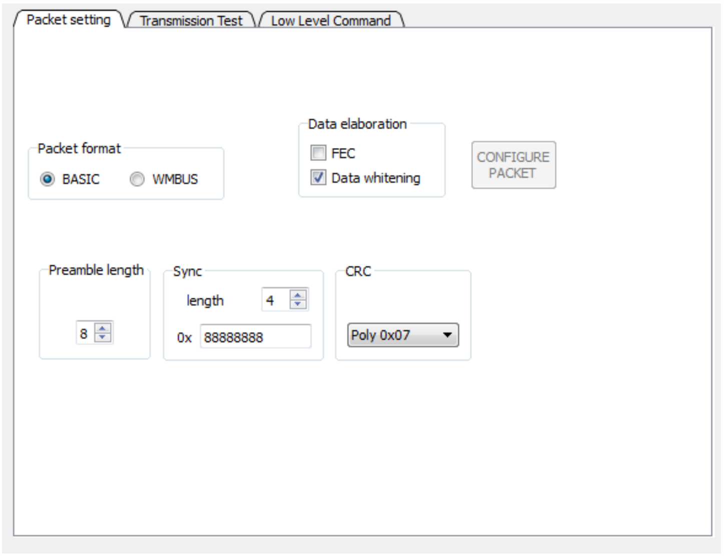
Selecting the "Panel setting" view, packet configurations are available to the user.

First of all, the desired packet format has to be selected 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 21. Packet setting panel (Basic packet view)



Packet format: [✓ BASIC] WMBUS

Data elaboration: FEC ☐, Data whitening ☑

Preamble length: 8

Sync length: 4

Sync: 0x 88888888

CRC: Poly 0x07

3.2.7 Packet setting - BASIC

As shown in *Figure 22*, selecting BASIC (default configuration), SPIRIT1 DK - DB uses the packet format BASIC described in the datasheet. The options for the packet are:

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

These entire fields can be changed according to these limits:

- Preamble length in the interval: [1 - 32] bytes.
- Sync length in the interval: [1 - 4] bytes.
- CRC:
  - NO CRC.
  - Poly 0x07 (1 byte).
  - Poly 0x8005 (2 bytes).
  - Poly 0x1021 (2 bytes).
  - Poly 0x864CFB (3 bytes).

The "FEC" and the "Data whitening" can be checked or not according to the desired setting. In particular, if the "FEC" is checked, this feature is used during the transmission; the same apply for "Data whitening".

3.2.8 Packet setting - WMBUS

As shown in *Figure 23*, selecting MBUS, SPIRIT1 DK - DB uses the packet format MBUS described in the datasheet. The options for the packet are:

- MBUS submode.
- Preamble length.
- Postamble length.

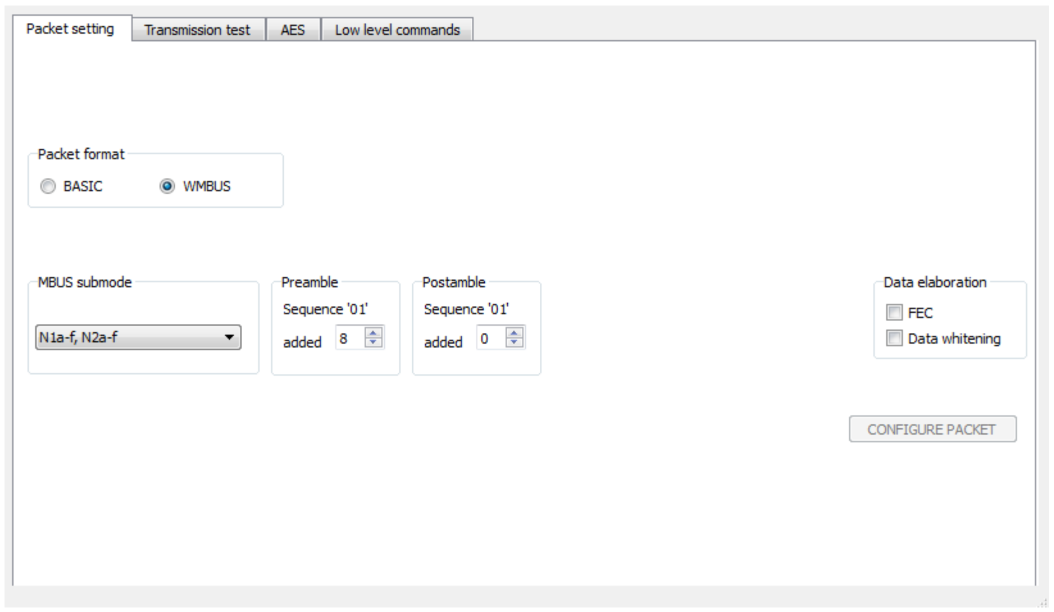
These entire fields can be changed according to these limits:

- Preamble length in the interval: [0 - 255] chip sequence (01).
- Postamble length in the interval: [0 - 255] chip sequence (01).
- MBUS submode:
  - S1, S2, long header.
  - S1m, S2, T2 other to meter.
  - T1, T2, meter to other.
  - R2 short header.
  - N1a-f, N2a-f

The "FEC" and the "Data whitening" can be checked or not according to the desired setting. In particular, if the "FEC" is checked, this feature is used during the transmission; the same apply for "Data whitening".

Using this packet setting, the transmitter builds the frame with a length field of one byte, and then adds the data blocks with CRC included. The GUI shows the frame received with the length field and the CRC field for each data blocks.

Figure 22. Packet setting panel (MBUS Packet view)



Packet setting | Transmission test | AES | Low level commands

Packet format: ○ BASIC [✓ WMBUS]

MBUS submode: N1a-f, N2a-f

Preamble: Sequence '01' added 8

Postamble: Sequence '01' added 0

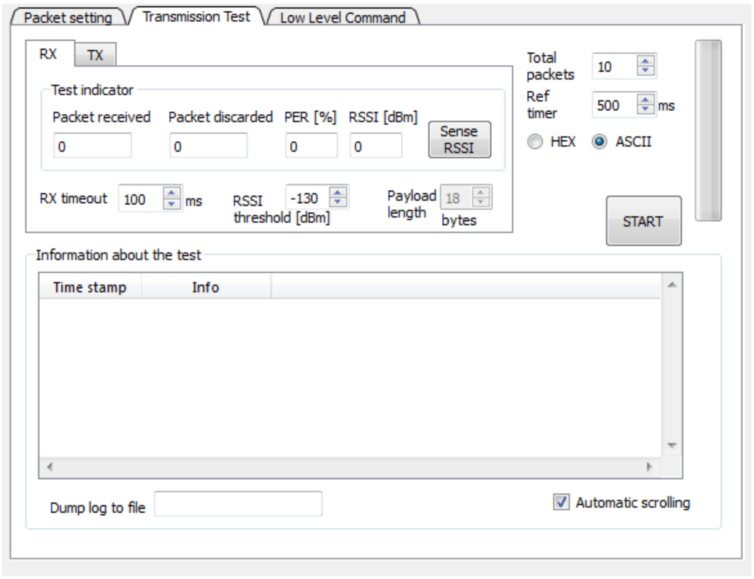
Data elaboration: ☐ FEC ☐ Data whitening

CONFIGURE PACKET

3.3 Transmission test

Selecting the "Transmission test" view, the user has all the controls to access the packet tests available until now to run transmission

Figure 23. Transmission test panel



Packet setting | Transmission Test | Low Level Command

RX | TX

Test indicator

Packet received	Packet discarded	PER [%]	RSSI [dBm]
0	0	0	0

Sense RSSI

RX timeout 100 ms

RSSI threshold [dBm] -130

Payload length 18 bytes

Total packets 10

Ref timer 500 ms

○ HEX [✓ ASCII]

START

Information about the test

Time stamp	Info

Dump log to file

☑ Automatic scrolling

On the left corner, it is possible to set the device main role during the transmission: RX and TX, in "Device role" panel.

The panel "Data to send" has the value expressed in hexadecimal or in characters that a transmitter sends. The max length of this field is 255 bytes (GUI arbitrary limitation, not device limitation), and represents the effective payload sent. If the HEX check box is checked, the value must be added in this way: 07 08 09 0A and so on, if the ASCII check box is checked, characters are accepted. There is also the possibility to generate a random set of values checking RAND and write how many bytes send in Payload length box. In this case, the random sequence has a length equal to the set one. Since it is not sure that the randomly generated chars can be converted to ASCII, they are provided in hex format.

In "RX timeout" box the RX timeout in milliseconds should be set: this value has to be set large enough to receive completely the SYNC word and have to be set according to the data rate, the preamble length and the sync length. The GUI computes automatically a value, but it can be changed. This period is necessary only during the RX state to allow the SPIRIT1 find correctly a SYNC word. If the value is 0, then the RX timeout is infinite, and the SPIRIT1 stays in RX state until it found a correct SYNC word. The data received can be displayed in HEX or ASCII. However, if ASCII is set and a non ASCII char is received, they will switch to hex format.

The SPIRIT1 has two modes of packet length: fix mode length and variable mode length. The former is used in this GUI and needs the receiver to know the length of the expected packet, in the latter instead the receiver find out the length of the packet from the packet itself, because the transmitter has filled a field of the frame with this information. To set the VAR mode (default), the bit 0 of the register PCKTCTRL2(0x32) must be set. To set FIX mode, this bit must be reset and the GUI will disable the length field on the RX tab. This operation can be easily done by the register table tab.

The panel "test indicator" shows all the results about the transmission/reception operations: the number of packet received correctly; the packet lost (also the packet lost for RX timeout that means the SPIRIT1 does not receive a packet within the RX timeout period); the RSSI value of the last packet; the PER from the start of the communication up to now.

On the right side, there are two controls: "Total packets" and "Ref timer". The former sets how many packets the transmitter will send or how many packets are expected by the receiver, an infinite number of packets can be sent if the value is 0; the latter sets the period of time in which a packet is sent.

The test implemented in the firmware of the GUI defines a cycle in which the SPIRIT1 configured as transmitter send a packet (the duration of this operation depends by the data rate and the approximate value is reported in the packet duration box) then the SPIRIT1 goes idle state named "READY" until the period set in the packet rate box expires; then the cycle is repeated. From the receiver side, the test works slightly the same: the SPIRIT1 goes in RX state a couple of milliseconds before the transmitter goes in TX state, then wait for the SYNC of the packet for the time write in RX timeout box, then if the packet is received or the RX timeout expires the SPIRIT1 goes the idle state (READY) until the period set in the packet rate box expires. Also, during first communication the SPIRIT1 goes in RX state waiting for the first packet (synchronization packet) with infinite RX timeout.

It is crucial to set the value of the packet rate greater than packet duration. Otherwise, to the received packet can be truncated. Also, the packet rate must be the same for both devices.

The "START" button makes the test run.

Once the test is started, the "START" button label is changed to "STOP" button. If this button is pressed while the test runs, the test is stopped.

In the bottom of the Transmission test panel, there is a box in which it is possible to write the name of a file in which the GUI saves a log of the current test. This operation is performed during the test, so it is important to write the filename before the test starts.

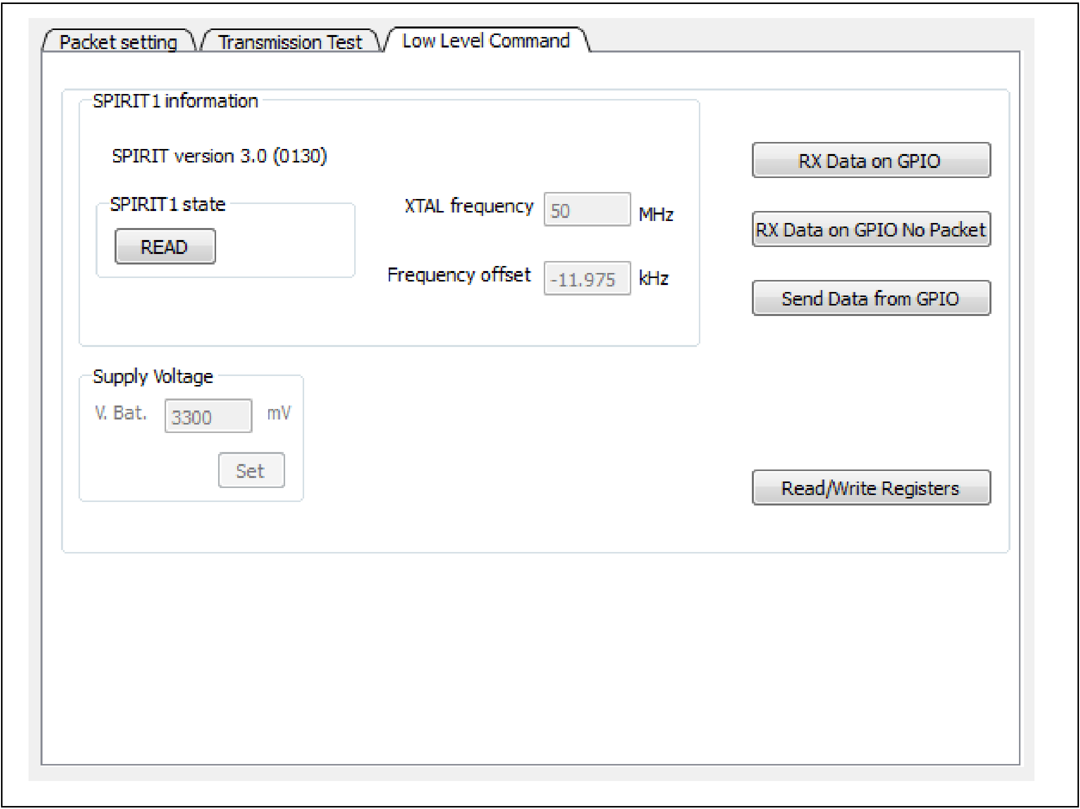
The button "Sense RSSI" can be used to read the RF power on air according to the center frequency and the channel filter bandwidth configured. This feature also allows knowing the RF noise in the environment.

The "RSSI threshold [dBm]" sets the threshold of the RSSI. To achieve a good communication, it is important to set in the receiver an RSSI threshold greater than the RX noise floor.

3.3.1 Low level commands

Selecting the "Low level commands" view, the user has all the controls to access the registers of the SPIRIT1.

Figure 24. Low level commands panel



With this tab, the user can use some low level feature that the GUI exports.

The SPIRIT1 status can be read by pressing the "SPIRIT1 state" READ button. Also, the version of the SPIRIT1 is given.

The XTAL frequency is available on the "XTAL frequency" textbox. On the boards provided with EEPROM, the frequency offset is also available. The user can use this parameter to correct the nominal center frequency in order to obtain a well-centered signal.

The three buttons allow setting some particular test modes:

- RX Data on GPIO
- Send Data from GPIO
- RX Data on GPIO No Packet

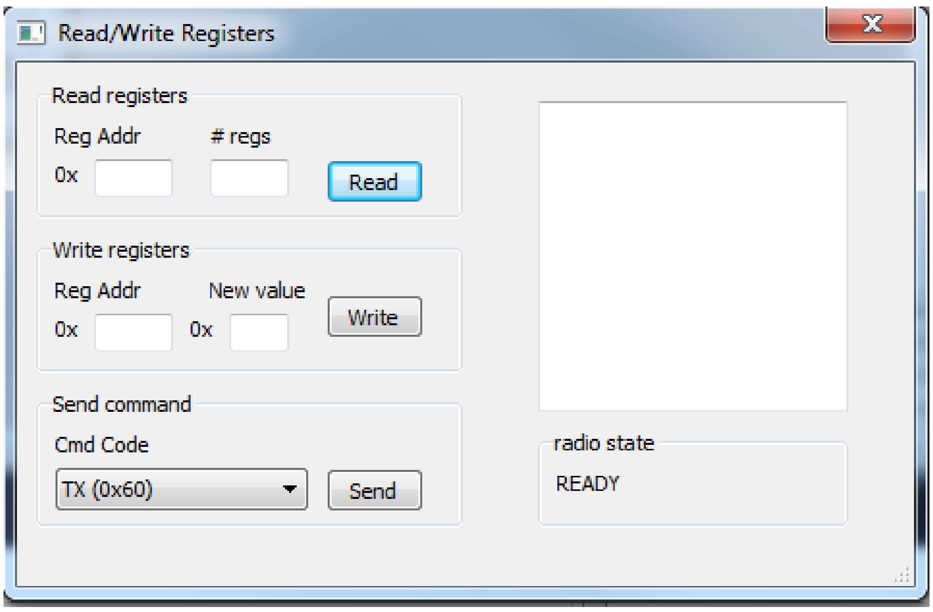
The RX Data on GPIO configures the GPIO_0 and the GPIO_1 of the SPIRIT1 to send respectively the RX data received and the clock signal. In this way, when the SPIRIT1 goes in RX state it is possible to see the packet received.

The Send Data from GPIO configures the GPIO_0 and the GPIO_1 of the SPIRIT1 to send respectively the data to transmit and the clock to sample the data. In this way, when the SPIRIT1 goes in TX state it is possible to send data loaded through the GPIO (and not through the FIFO as usual).

The RX Data on GPIO No Packet is equal to the RX Data on GPIO with the only difference that the packet handler embedded in the SPIRIT1 is by-passed.

While through the registers table, it is possible to write the most used registers, the button Read/Write Registers allows to write all SPIRIT1 registers. If clicked, the following window opens:

Figure 25. Low level register setting



From here it is possible to specify:

- In case of write, the address and the value of one single register

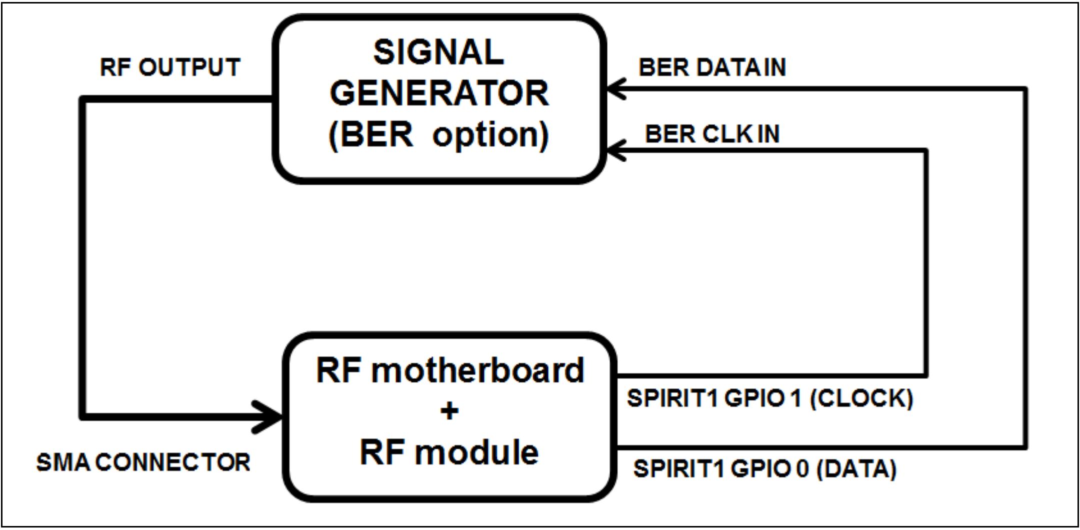
- In the event of read, the starting address and the number of registers to be read from there

Finally, the group Supply Voltage. It can be used to regulate the VBAT of the device specifying the desired voltage in mV. This function is available only for STEVAL-IKR001Vx - RF motherboard. Otherwise, it is disabled.

3.3.2 How to run a BER test using a signal generator

Through the low level commands tab, it is possible to put the SPIRIT1 in RX direct more through GPIOs. In this mode, the packet handler is entirely bypassed, and the demodulated data plus associated clock signal is available on two GPIOs. This mode can be enabled by the button "RX Data on GPIO No Packet". The two signals then can be used as in Figure LL in a signal generator with BER option to allow measuring the Bit Error Rate according to that particular radio configuration. The data must be sampled on falling edge of the clock signal.

Figure 26. BER test using SPIRIT1 DK - GUI

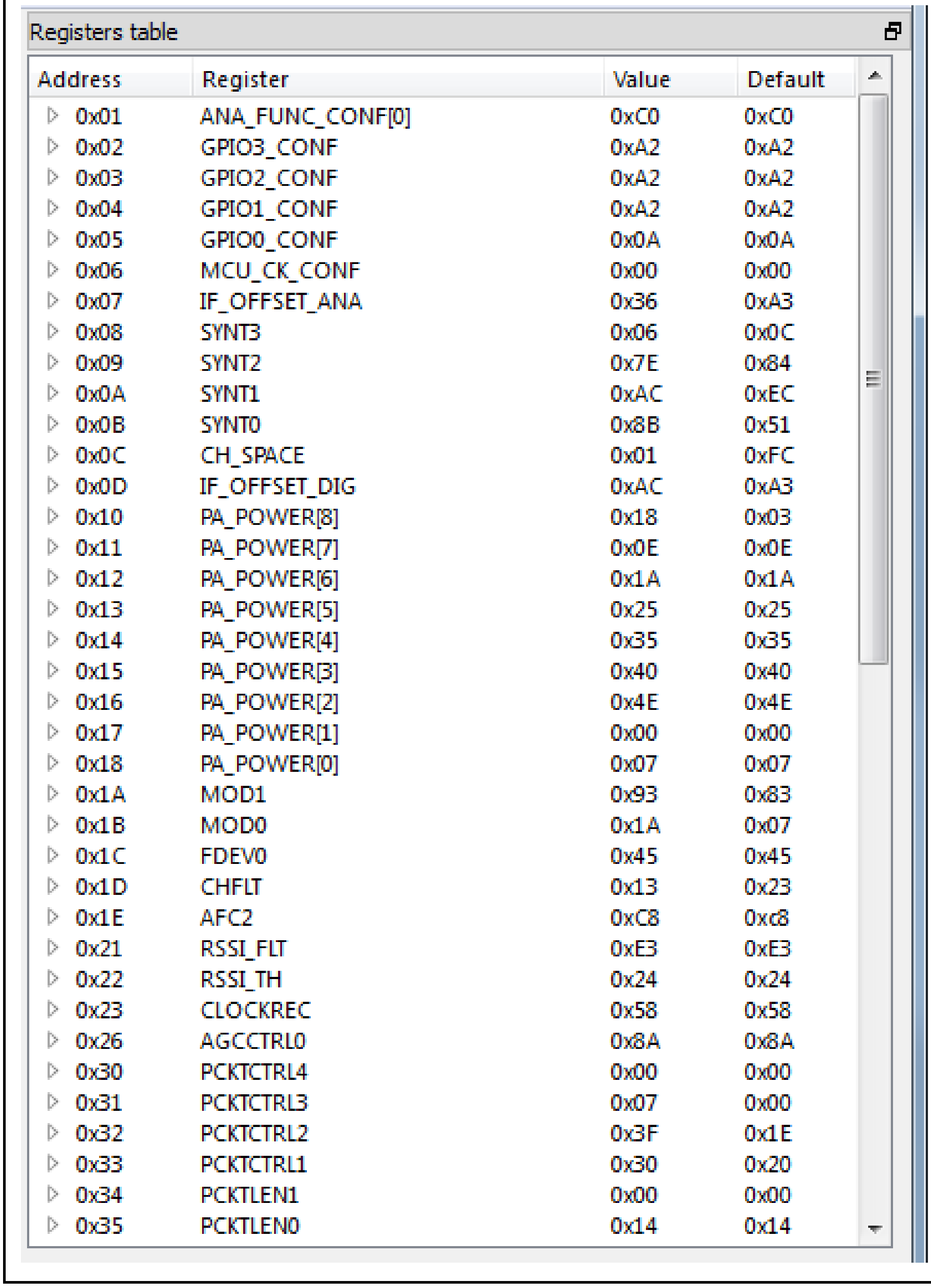


3.4 Registers table

On the right side of the GUI, a register table is shown by default (it can be hide/show using the “<<” button).

The table (ref. *Figure 27*) provides a quick and user-friendly way to modify the registers of the device and its bit-fields.

Figure 27. Registers table

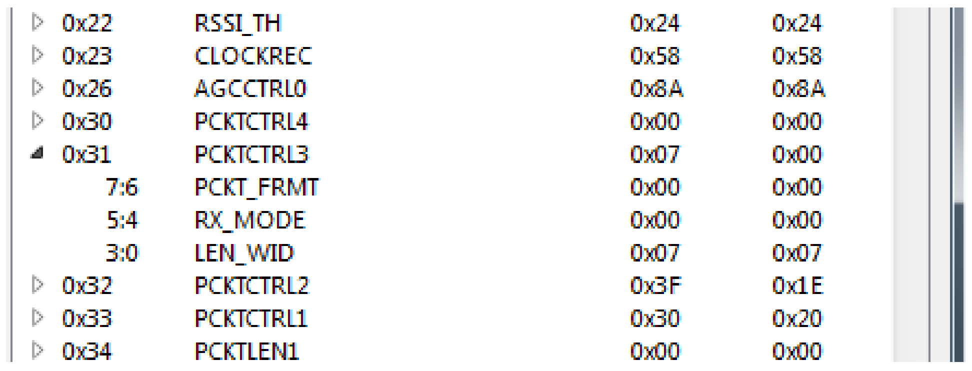


Registers table

Address	Register	Value	Default
0x01	ANA_FUNC_CONF[0]	0xC0	0xC0
0x02	GPIO3_CONF	0xA2	0xA2
0x03	GPIO2_CONF	0xA2	0xA2
0x04	GPIO1_CONF	0xA2	0xA2
0x05	GPIO0_CONF	0x0A	0x0A
0x06	MCU_CK_CONF	0x00	0x00
0x07	IF_OFFSET_ANA	0x36	0xA3
0x08	SYNT3	0x06	0x0C
0x09	SYNT2	0x7E	0x84
0x0A	SYNT1	0xAC	0xEC
0x0B	SYNT0	0x8B	0x51
0x0C	CH_SPACE	0x01	0xFC
0x0D	IF_OFFSET_DIG	0xAC	0xA3
0x10	PA_POWER[8]	0x18	0x03
0x11	PA_POWER[7]	0x0E	0x0E
0x12	PA_POWER[6]	0x1A	0x1A
0x13	PA_POWER[5]	0x25	0x25
0x14	PA_POWER[4]	0x35	0x35
0x15	PA_POWER[3]	0x40	0x40
0x16	PA_POWER[2]	0x4E	0x4E
0x17	PA_POWER[1]	0x00	0x00
0x18	PA_POWER[0]	0x07	0x07
0x1A	MOD1	0x93	0x83
0x1B	MOD0	0x1A	0x07
0x1C	FDEV0	0x45	0x45
0x1D	CHFLT	0x13	0x23
0x1E	AFC2	0xC8	0xc8
0x21	RSSI_FLT	0xE3	0xE3
0x22	RSSI_TH	0x24	0x24
0x23	CLOCKREC	0x58	0x58
0x26	AGCCTRL0	0x8A	0x8A
0x30	PCKTCTRL4	0x00	0x00
0x31	PCKTCTRL3	0x07	0x00
0x32	PCKTCTRL2	0x3F	0x1E
0x33	PCKTCTRL1	0x30	0x20
0x34	PCKTLEN1	0x00	0x00
0x35	PCKTLEN0	0x14	0x14

The single register can be expanded or compressed to show the logical fields that compose it. This is done by clicking on the white arrow to the left of each entry.

Figure 28. Registers table



▷ 0x22	RSSI_TH	0x24	0x24
▷ 0x23	CLOCKREC	0x58	0x58
▷ 0x26	AGCCTRL0	0x8A	0x8A
▷ 0x30	PCKTCTRL4	0x00	0x00
◢ 0x31	PCKTCTRL3	0x07	0x00
7:6	PCKT_FRMT	0x00	0x00
5:4	RX_MODE	0x00	0x00
3:0	LEN_WID	0x07	0x07
▷ 0x32	PCKTCTRL2	0x3F	0x1E
▷ 0x33	PCKTCTRL1	0x30	0x20
▷ 0x34	PCKTLEN1	0x00	0x00

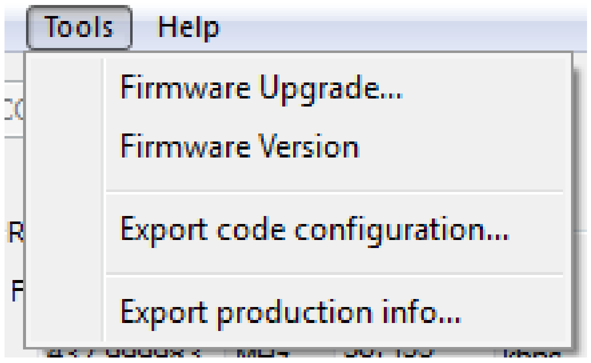
When a field is modified, the correspondent register is automatically written in spirit1. Moreover, if the register modifies a parameter of the radio part or packet, the correspondent tab is updated with the new field value. It is also possible to get a detailed description of a register double clicking on its entry in the registers table.

Five buttons are available on the bottom side of the tab:

- Refresh: reads all the registers 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 to a file selected by the user the register configuration
- Import: loads the registers from a file selected by the user. The file can be loaded both in XML and txt (with a notation like address: value).

3.5 Tools

Figure 29. Tools list



Opening the tools list a set of options are available to the user to check and update firmware and also to load and save radio configurations.

3.5.1 Firmware upgrade

The SPIRIT1 DK is shipped with a firmware which allows performing automatic firmware upgrade via the USB port. The default firmware is composed of two applications:

- DFU boot loader which allows to perform firmware upgrade
- Application firmware that runs the application selected by the user

In order to perform firmware upgrade the user should:

1. Activate the DFU software manually, by pressing and releasing "RESET" button while holding down the "Push_button" (the button is the SCM_PS for the STEVAL-IKR001Vx and for the STEVAL-IDS001Vx board the procedure is unplug the USB RF dongle, press and hold the button SW1 and then plug the USB RF dongle and release the SW button). The board LED DL2 (LED D2 in the STEVAL-IDS001Vx boards) will start flashing to confirm that DFU boot loader is running.
2. Launch the SPIRIT1 DK GUI and from Tools->Firmware Upgrade select the firmware image to load
3. After clicking OK the firmware is programmed into the board.

The firmware images are in Intel hex format and are stored in the folder <Installation Path>\Firmware\Binary.

The firmware images present in the folder are described in *Table 4*.

Table 4. SPIRIT DK firmware images

Firmware image name	Description
SPIRIT1_CLI.hex	SPIRIT1 CLI firmware. CLI stands for command line and it can be used in conjunction with SPIRIT1 DK GUI
WMBUS_SND_IR-meter.hex	SPIRIT1 WM-BUS meter demo firmware image.
WMBUS_GUI.hex	SPIRIT1 WM-BUS concentrator GUI demo firmware image.
SPIRIT1_DK-DFU.hex	SPIRIT1 DFU boot loader image. This is not suitable for automatic firmware upgrade and it is provided only for crash recovery if the DFU boot loader has been deleted by mistake.

3.5.2 Firmware version

The Firmware Version can be used to see the version of the current firmware running in the microcontroller.

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

It is important to notice that 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.

3.5.3 Save and load high level configurations

The Save option allows to save the current radio configuration and packet configuration in a file, in order to make easy to restore it at later time.

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

A set of predefined radio configuration are provided in the default folder <Installation Path>\Spirit_GUI_Configuration.

Save and load registers configurations

These features are available by the buttons "Export" and "Import" below the registers table. Please see section 4.2.7 for further details.

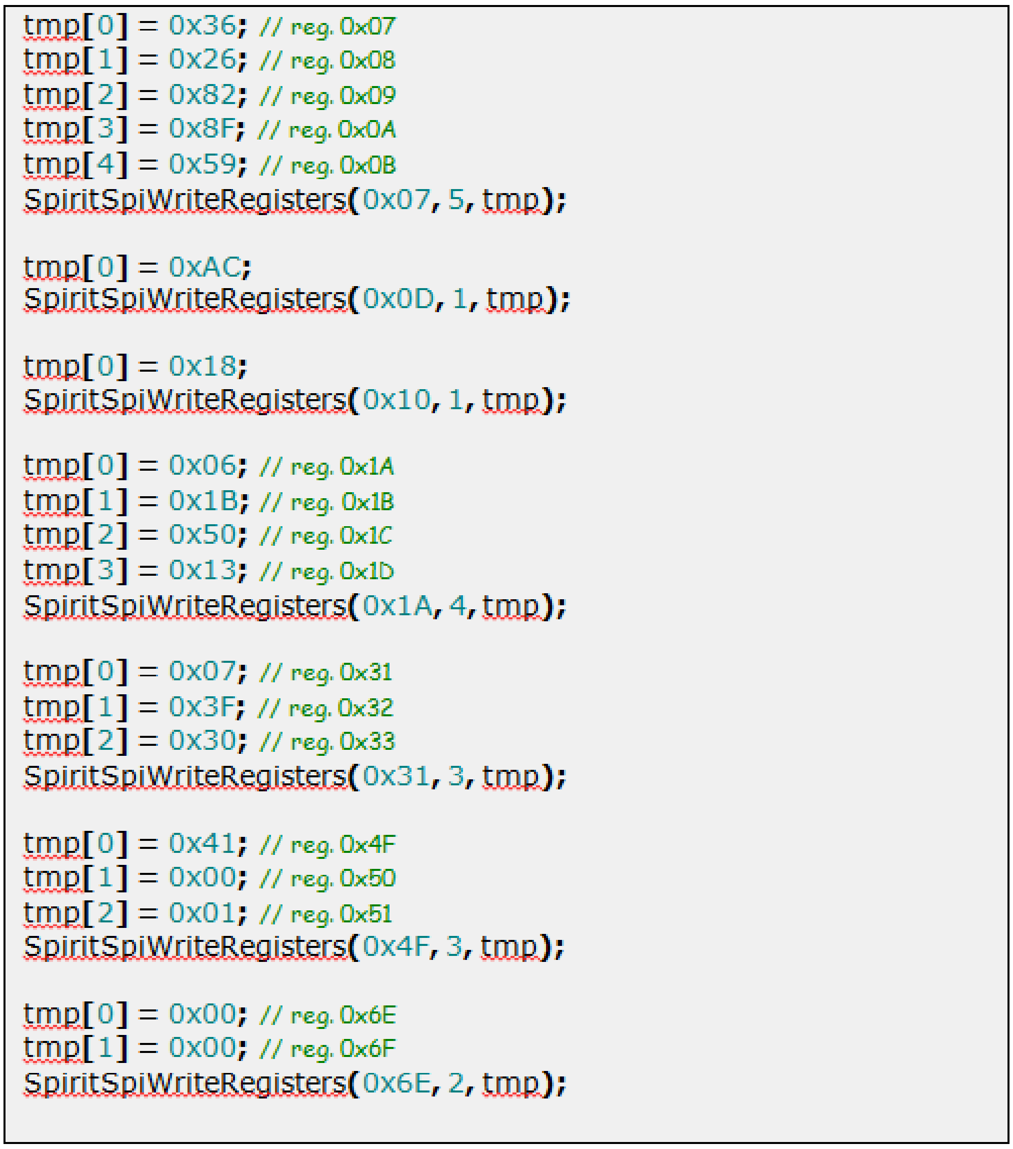
Export code configuration

This option generates a C-language list of instructions that write on the SPIRIT1 registers the values that are different from the default values.

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

An example of what is obtained as an output file follows. To guarantee the maximum freedom to the user, it is just a list of instructions. According to the application, it can be modified manually to become a function, a macro or a simple block of instructions.

Figure 30. Example of C code exported from the GUI

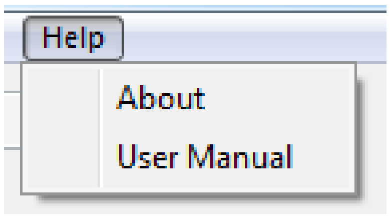


```
tmp[0] = 0x36; // reg. 0x07
tmp[1] = 0x26; // reg. 0x08
tmp[2] = 0x82; // reg. 0x09
tmp[3] = 0x8F; // reg. 0x0A
tmp[4] = 0x59; // reg. 0x0B
SpiritSpiWriteRegisters(0x07, 5, tmp);

tmp[0] = 0xAC;
SpiritSpiWriteRegisters(0x0D, 1, tmp);

tmp[0] = 0x18;
SpiritSpiWriteRegisters(0x10, 1, tmp);

tmp[0] = 0x06; // reg. 0x1A
tmp[1] = 0x1B; // reg. 0x1B
tmp[2] = 0x50; // reg. 0x1C
tmp[3] = 0x13; // reg. 0x1D
SpiritSpiWriteRegisters(0x1A, 4, tmp);

tmp[0] = 0x07; // reg. 0x31
tmp[1] = 0x3F; // reg. 0x32
tmp[2] = 0x30; // reg. 0x33
SpiritSpiWriteRegisters(0x31, 3, tmp);

tmp[0] = 0x41; // reg. 0x4F
tmp[1] = 0x00; // reg. 0x50
tmp[2] = 0x01; // reg. 0x51
SpiritSpiWriteRegisters(0x4F, 3, tmp);

tmp[0] = 0x00; // reg. 0x6E
tmp[1] = 0x00; // reg. 0x6F
SpiritSpiWriteRegisters(0x6E, 2, tmp);
```

3.5.4 Help

Opening the Help list a link to the user manual is available.

Figure 31. Help with user manual

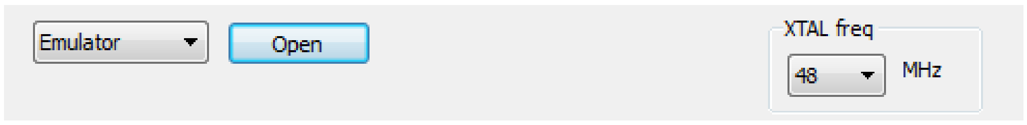


3.5.5 Device emulator

Another possibility, that the user has, is to simulate the device without any board connected to the PC.

An entry always present is indeed "Emulator". When it is selected, the user can use the GUI as if a board was connected to the PC.

Figure 32. Emulator port



Since a board is not really present, the user should specify manually a XTAL frequency (that is otherwise computed automatically by the microcontroller) using the proper tab that is active in this case only.

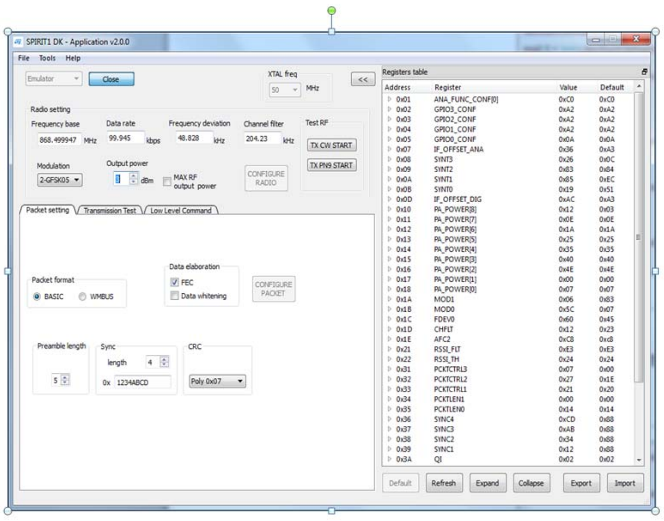
From there on, clicking the button "Open" everything related to the device config should run exactly as if a device is connected.

The intent is to allow the user to select his own configuration easily and then see or save the register values that are needed in order to keep the same configuration on his firmware.

For this purpose, this feature can be used in cooperation with the "Export code configuration" tool (see section 4.2.8.5).

Hereafter an example of what is obtained when the configuration on the left is exported as the .c file on the right.

Figure 33. GUI vs. registers C-code



XTAL freq: 50 MHz

Radio setting

Frequency base: 868.499947 MHz

Data rate: 99.945 kbps

Frequency deviation: 48.828 kHz

Channel filter: 204.23 kHz

Test RF

Modulation: 2-GFSK05

Output power: [illegible] dBm

[ ] MAX RF output power

Packet format: [✓ BASIC] WMBUS

Data elaboration: [✓] FEC [ ] Data whitening

Preamble length: 5

Sync length: 4

Sync: 0x 1234ABCD

CRC: Poly 0x07

Registers table

Address	Register	Value	Default
0x01	ANA_FUNC_CONF[0]	0xC0	0xC0
0x02	GPIO3_CONF	0xA2	0xA2
0x03	GPIO2_CONF	0xA2	0xA2
0x04	GPIO1_CONF	0xA2	0xA2
0x05	GPIO0_CONF	0x0A	0x0A
0x07	IF_OFFSET_ANA	0x36	0xA3
0x08	SYNT3	0x26	0x0C
0x09	SYNT2	0x83	0x84
0x0A	SYNT1	0x85	0xEC
0x0B	SYNT0	0x19	0x51
0x0D	IF_OFFSET_DIG	0xAC	0xA3
0x10	PA_POWER[8]	0x12	0x03
0x11	PA_POWER[7]	0x0E	0x0E
0x12	PA_POWER[6]	0x1A	0x1A
0x13	PA_POWER[5]	0x25	0x25
0x14	PA_POWER[4]	0x35	0x35
0x15	PA_POWER[3]	0x40	0x40
0x16	PA_POWER[2]	0x4E	0x4E
0x17	PA_POWER[1]	0x00	0x00
0x18	PA_POWER[0]	0x07	0x07
0x1A	MOD1	0x06	0x83
0x1B	MOD0	0x5C	0x07
0x1C	FDEV0	0x60	0x45
0x1D	CHFLT	0x12	0x23
0x1E	AFC2	0xC8	0xc8
0x21	RSSI_FLT	0xE3	0xE3
0x22	RSSI_TH	0x24	0x24
0x31	PCKTCTRL3	0x07	0x00
0x32	PCKTCTRL2	0x27	0x1E
0x33	PCKTCTRL1	0x21	0x20
0x34	PCKTLEN1	0x00	0x00
0x35	PCKTLEN0	0x14	0x14
0x36	SYNC4	0xCD	0x88
0x37	SYNC3	0xAB	0x88
0x38	SYNC2	0x34	0x88
0x39	SYNC1	0x12	0x88
0x3A	QI	0x02	0x02

```c
tmp[0] = 0x36; // reg. 0x07
tmp[1] = 0x26; // reg. 0x08
tmp[2] = 0x83; // reg. 0x09
tmp[3] = 0x85; // reg. 0x0A
tmp[4] = 0x19; // reg. 0x0B
SpiritSpiWriteRegisters(0x07, 5, tmp);

tmp[0] = 0xAC;
SpiritSpiWriteRegisters(0x0D, 1, tmp);

tmp[0] = 0x12;
SpiritSpiWriteRegisters(0x10, 1, tmp);

tmp[0] = 0x06; // reg. 0x1A
tmp[1] = 0x5C; // reg. 0x1B
tmp[2] = 0x60; // reg. 0x1C
tmp[3] = 0x12; // reg. 0x1D
SpiritSpiWriteRegisters(0x1A, 4, tmp);

tmp[0] = 0x07; // reg. 0x31
tmp[1] = 0x27; // reg. 0x32
tmp[2] = 0x21; // reg. 0x33
SpiritSpiWriteRegisters(0x31, 3, tmp);

tmp[0] = 0xCD; // reg. 0x36
tmp[1] = 0xAB; // reg. 0x37
tmp[2] = 0x34; // reg. 0x38
tmp[3] = 0x12; // reg. 0x39
SpiritSpiWriteRegisters(0x36, 4, tmp);

tmp[0] = 0x41; // reg. 0x4F
tmp[1] = 0x00; // reg. 0x50
tmp[2] = 0x01; // reg. 0x51
SpiritSpiWriteRegisters(0x4F, 3, tmp);

tmp[0] = 0x00; // reg. 0x6E
tmp[1] = 0x00; // reg. 0x6F
SpiritSpiWriteRegisters(0x6E, 2, tmp);
```

4 SPIRIT1 WM-BUS GUI description

This section describes how to use the Spirit1 Wireless M-BUS-GUI.

4.1 Firmware installation

The boards come preprogrammed with firmware suitable to run Spirit1 DK-GUI, in order to operate the WM-BUS GUI, it is necessary to reprogram the appropriate firmware. Following the instructions in section 3.5.1 and load the firmware image Firmware/Binary/WMBUS_GUI.hex to load the GUI firmware on one board.

The other board should be programmed with a meter example from the Firmware/STM32L/WMBUS_Applications_2013/WMBUS/IAR .

The user can configure the WMBUS submode of these examples editing the user_config.h file.

For simplicity, a pre-compiled hex file for the meter is located in the Binary folder: Firmware/Binary/WMBUS_SND_IR-meter.hex . The example is precompiled with a submode that is the same of the GUI default.

4.2 Detailed description

The SPIRIT1 Wireless M-BUS - GUI requires only the concentrator to be connected to the PC, the meter is operated through joystick and buttons, and it needs USB connection just for power.

After the boards are prepared as described in section 4.1, the user can run the GUI. After running the GUI, the following steps needs to be performed:

1. Connect the concentrator board to the PC.
2. Click the connect button, you should get the message "Connection Successful" (see *Figure 35*)
3. Select the "Configure" tab and click on “Read all” (see *Figure 38*)
4. Please check the parameters and in particular that "Device type" is concentrator and WM-Bus mode is S2 or in general, the same as the meter(*Figure 37*).
5. Go to the Meter DB tab and click on the init Database button to initialize the meters database for the first time. It is also possible to modify the meter properties.
6. Go to the Meter Data tab and click Start.
7. Power the meter board and use the joystick as explained in the <example>_README.txt (located in the same folder as the application .c file) to initiate a communication. The packets, together with their link layer level fields, are displayed in the "Meter Data" Tab. (*Figure 38*).

Figure 34. WM-BUS GUI step 2

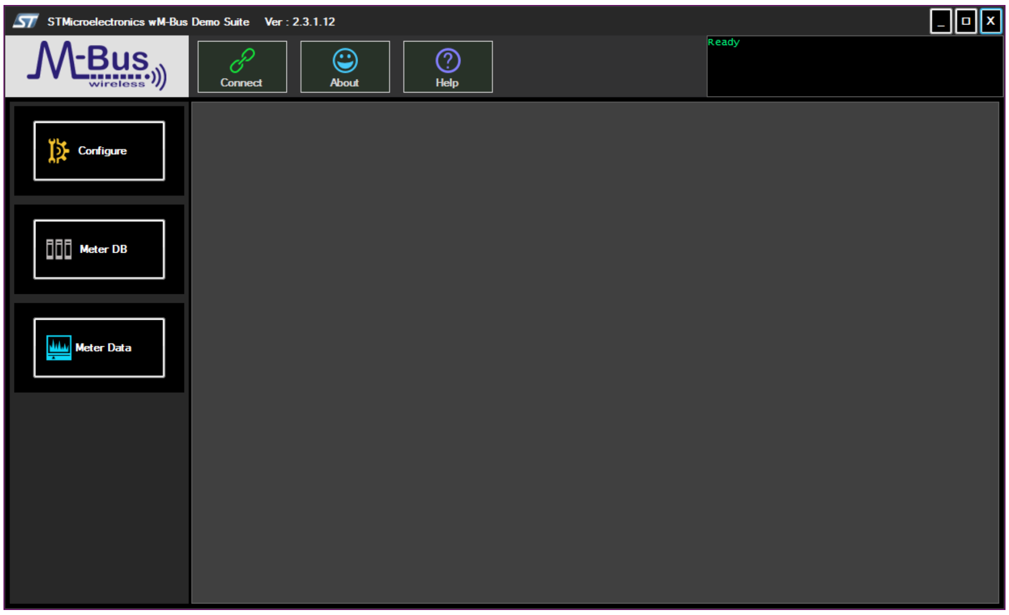
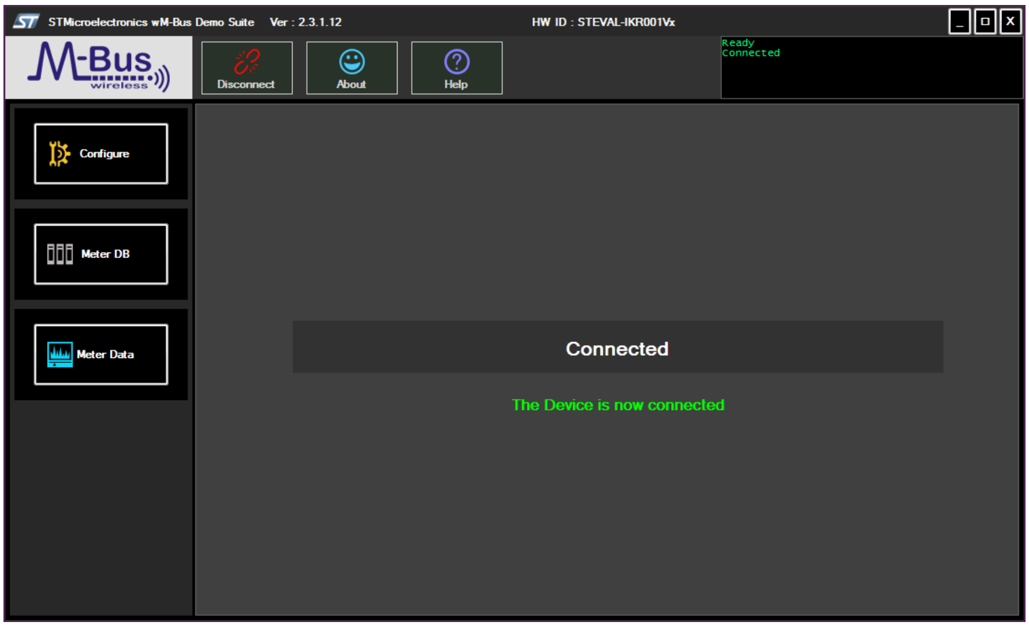


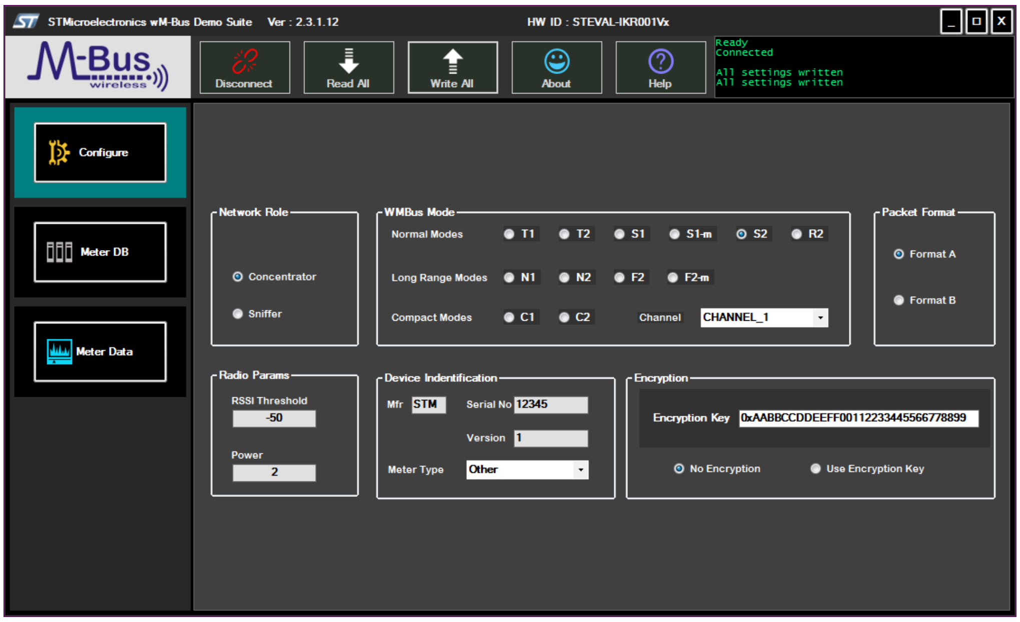
Figure 35. WM-BUS GUI step 2 (successful connection)



Connected

The Device is now connected

Figure 36. WM-BUS GUI step 3 and 4



```
Ready
Connected

All settings written
All settings written
```

Network Role: [✓ Concentrator] Sniffer

WMBus Mode:
Normal Modes: T1 T2 S1 S1-m [✓ S2] R2
Long Range Modes: N1 N2 F2 F2-m
Compact Modes: C1 C2
Channel: CHANNEL_1

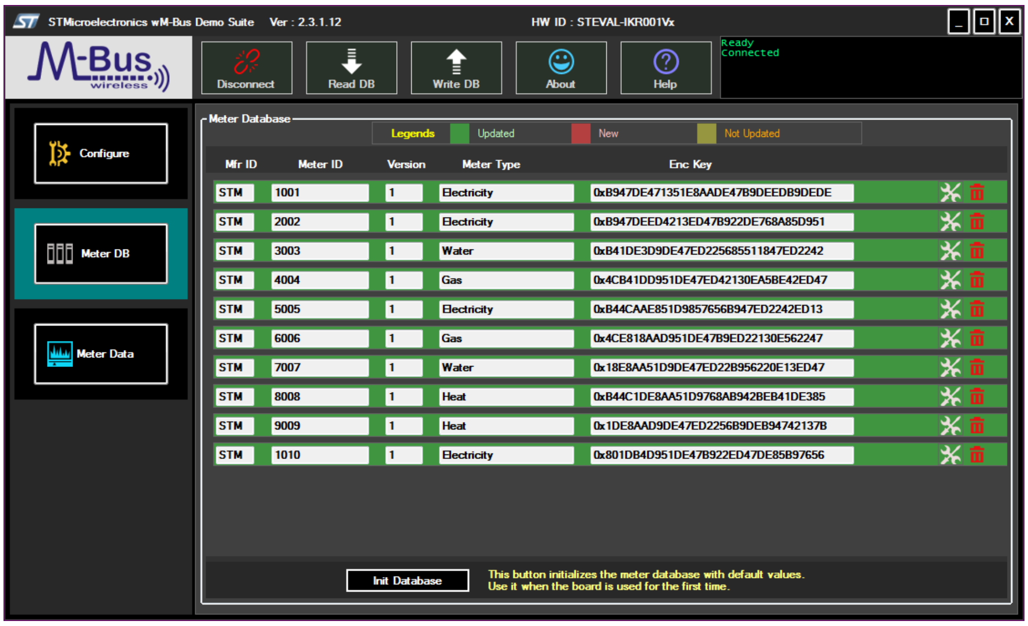
Packet Format: [✓ Format A] Format B

Radio Params:
RSSI Threshold: -50
Power: 2

Device Indentification:
Mfr: STM
Serial No: 12345
Version: 1
Meter Type: Other

Encryption:
Encryption Key: 0xAABBCCDDEEFF00112233445566778899
[✓ No Encryption] Use Encryption Key

Figure 37. WM-BUS GUI step 5



```
Ready
Connected
```

Meter Database

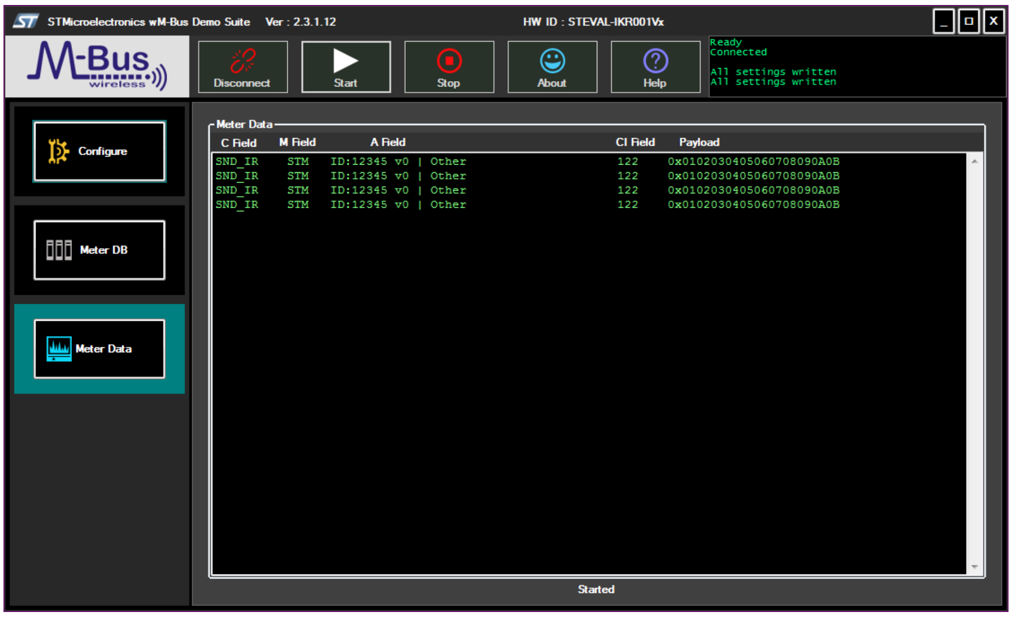
Legends: 🟩 Updated 🟥 New 🟨 Not Updated

Mfr ID	Meter ID	Version	Meter Type	Enc Key
STM	1001	1	Electricity	0xB947DE471351E8AADE47B9DEEDB9DEDE
STM	2002	1	Electricity	0xB947DEED4213ED47B922DE768A85D951
STM	3003	1	Water	0xB41DE3D9DE47ED225685511847ED2242
STM	4004	1	Gas	0x4CB41DD951DE47ED42130EA5BE42ED47
STM	5005	1	Electricity	0xB44CAAE851D9857656B947ED2242ED13
STM	6006	1	Gas	0x4CE818AAD951DE47B9ED22130E562247
STM	7007	1	Water	0x18E8AA51D9DE47ED22B956220E13ED47
STM	8008	1	Heat	0xB44C1DE8AA51D9768AB942BEB41DE385
STM	9009	1	Heat	0x1DE8AAD9DE47ED2256B9DEB94742137B
STM	1010	1	Electricity	0x801DB4D951DE47B922ED47DE85B97656

Init Database

This button initializes the meter database with default values.
Use it when the board is used for the first time.

Figure 38. WM-BUS GUI step 6, 7



```
Ready
Connected

All settings written
All settings written
```

Meter Data

C Field	M Field	A Field	CI Field	Payload
SND_IR	STM	ID:12345 v0 | Other	122	0x0102030405060708090A0B
SND_IR	STM	ID:12345 v0 | Other	122	0x0102030405060708090A0B
SND_IR	STM	ID:12345 v0 | Other	122	0x0102030405060708090A0B
SND_IR	STM	ID:12345 v0 | Other	122	0x0102030405060708090A0B

5 SPIRIT1 driver and example programs

The installed software package also contains support for SPIRIT1 firmware development.

The directory tree includes:

- SPIRIT1 driver in source form for STM32L
- Example IAR projects for STM32L in:
  - ...\Firmware\SPIRIT1_Library_Project\IAR\Spirit_Library_Project.eww
- Associated documentation in:
  - ...\Documents

In order to use or modify the example project, it is required to have installed IAR Embedded Workbench for ARM v6.50 and a JTAG interface like:

- ST-LINK

Or

- IAR JLink

6 SPIRIT1 wireless M-BUS library and example programs

The installed software package also contains support for SPIRIT1 Wireless M-BUS firmware development.

The directory tree includes:

- SPIRIT1 Wireless M-BUS library driver in binary form for STM32L
- Example IAR projects for STM32L in:
  - .....\Firmware\WMBUS_Example_STM32L\IAR\SPIRIT1-WMBUS_Example.eww
- Associated documentation in:
  - ...\Documents

In order to use or modify the example project, it is required to have installed IAR Embedded Workbench for ARM v6.50 and a JTAG interface like:

- ST-LINKv2

Or

- IAR JLink

7 Revision history

Table 5. Document revision history

Date	Revision	Changes
28-Nov-2013	1	Initial release.
04-Feb-2015	2	Minor text changes to improve readability. Added *Section 1.4*, *Section 1.6* Updated: *Section 1.3*, *Section 1.5*, *Chapter 2: Software installation*, *Chapter 3: SPIRIT1 DK - GUI description*; *Chapter 4: SPIRIT1 WM-BUS GUI description*; *Chapter 5: SPIRIT1 driver and example programs* and *Chapter 6: SPIRIT1 wireless M-BUS library and example programs*.
09-Dec-2015	3	Updated: Table titles in *Table 1: Motherboard MCU pin description versus function* and *Table 2: Dongle MCU pin description versus function*, *Section 3.4: Registers table* and *Section 4: SPIRIT1 WM-BUS GUI description*.