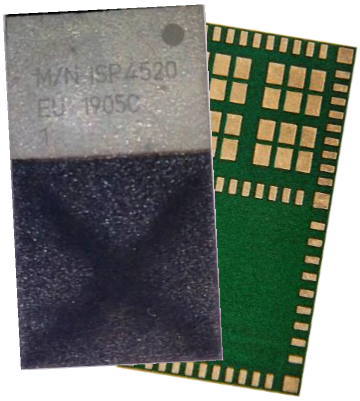
Use of ISP4520 Development Kit



Application Note AN190301

Introduction

Scope

This document gives details on hardware and software for using and testing Insight SiP Combo LoRa - BLE module ISP4520

Contents

1. Recommended Documentation

The following documents and Dev Kits (software portion) are required to understand the complete setup and programming methods:

Nordic Semiconductor Documents

- nRF52832 Development kit User Guide (hardware section should be partially ignored – Insight SiP development kit hardware replaces Nordic Semiconductor hardware).
- nRF52 Series Reference Manual.
- nRF52832 PS (data sheet).
- S132 nRF52832 SoftDevice Specification.
- nRF5 SDK (for software development on the nRF51 and nRF52 Series).

To access documentation, go to:

- Official Nordic Semi website *http://www.nordicsemi.com*
- The Nordic Semiconductor Documentation library *https://www.nordicsemi.com/DocLib*
- Ask any Nordic related question and get help *https://devzone.nordicsemi.com/questions*
- For any question, you can also open a case on the *http://www.nordicsemi.com*

Software Dev kits

- nRFgo Studio.
- nRF5 Software Development Kit (SDK) which includes precompiled HEX files, source code as well as SES and Keil ARM project files.
- SoftDevices for nRF52832.

To access these files, go to *www.nordicsemi.com* and download the files. Instructions can be found in Section 3.

Semtech Documents

- SX1261-SX1262 Product Datasheet
- SX1261 Calculator Tools
- Semtech Application Notes concerning SX1261/SX1262 and LoRa

To access documentation, information, go to *https://www.semtech.com/products/wireless-rf/lora-transceivers/sx1261*

Other Insight SiP documents

To complete the above, following documents are available on Insight SIP website or/and on request:

- AN190301 App Note – this document.
- DS4520 module data sheet.
- ISP4520-XX-GW Gateway Board schematic "ISP4520C_GW_SCH".
- ISP4520-XX-TB Test Board schematic "ISP4520C_TB_SCH"
- ISP130603 Interface Board schematic "SC130604".

2. ISP4520 Dev Kit Hardware Content

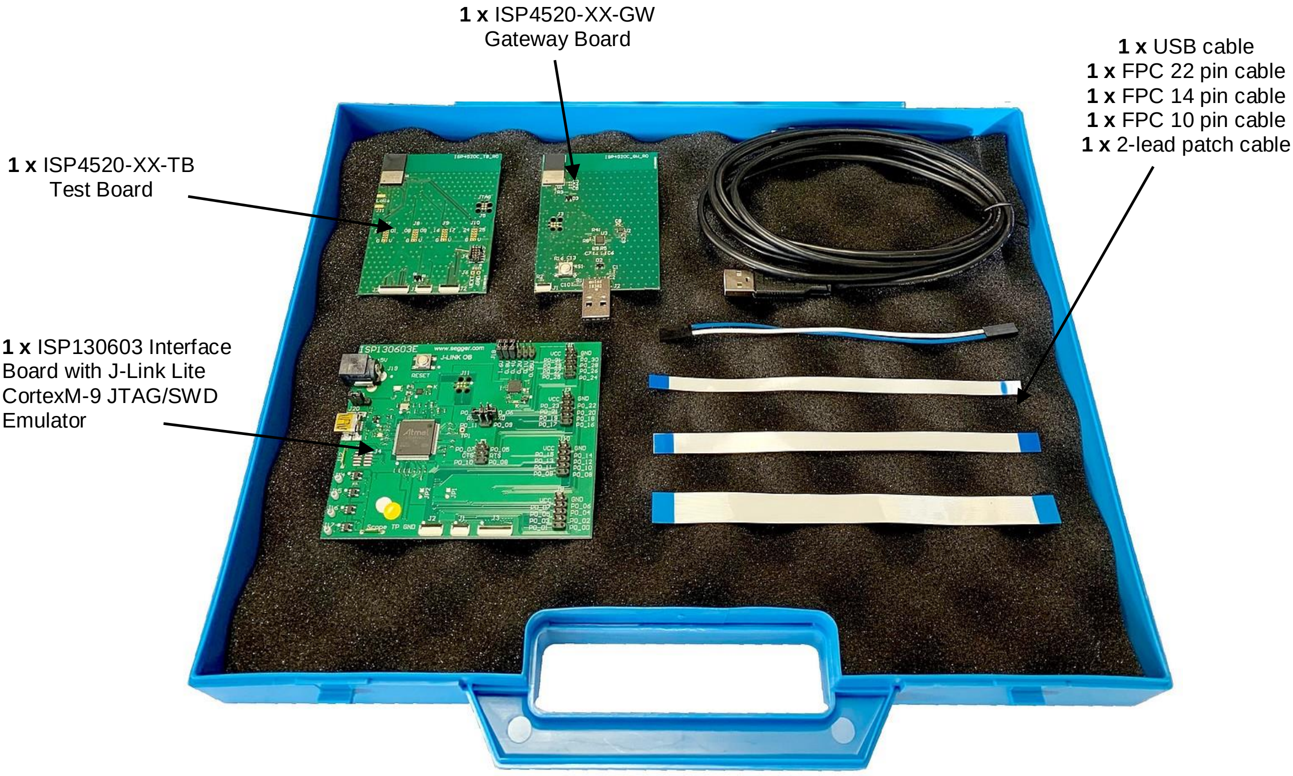


Figure 1: ISP4520 Dev Kit content

3. Software Installation

This paragraph describes the steps to follow for software installation.

1. Download and install the IDE of your choice:

   a. Download and install Keil MDK-ARM from *https://www.keil.com/demo/eval/arm.htm* to your hard drive. After installation, a Pack Installer window appears. Click on the "Packs" section and "Check for updates". After you can show in the Device section "Nordic Semiconductor" on the left side of the screen and different Packs available on the right side of the screen. On the "Packs" section, you can download and update Nordic example, nRF SoftDevice, nRF DeviceFamilyPack, nRF examples…, etc …Note that the free Keil MDK-ARM IDE is limited to 30k file size. For large applications a paid license is required.

   b. Download and install Segger Embedded Studio (SES) from *https://www.segger.com/downloads*. Go to *https://license.segger.com/Nordic.cgi* and fill the form. A free license key will be emailed to you, start SES, go to Tools -> License Manager and activate your license. This option allows compilation and loading of large applications.

2. Download and run the J-Link Software and documentation pack for Windows from *http://www.segger.com/jlink-software.html.*

3. Go to www.nordicsemi.com and log in to your Nordic My Page account.

Go to Products and click on Bluetooth Smart/Bluetooth Energy. You will have access to the different product:

   a. nRF52 Series: Click on nRF52832 and on the download section you have access to the documentation, SoftDevice, nRFgo studio, SDK … etc …

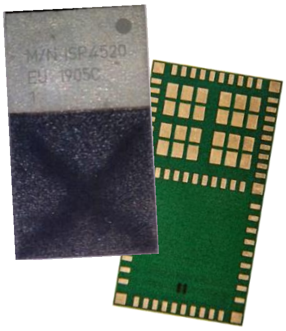
   b. nRF52 Development Tools: You can download the last nRF5 SDK.

4. You can also download the SDK in the following link: *https://developer.nordicsemi.com/*.

5. Download and install nRFgo Studio (Make sure to download the last version updated). Alternatively nRF Connect for Desktop (with the programmer plug-in) can be used for programming. Go to *https://www.nordicsemi.com/Software-and-Tools/Development-Tools/nRF-Connect-for-desktop*, download and install the latest version.

4. Hardware Description

4.1. ISP4520 Module



ISP4520 is a LoRa and Bluetooth Low Energy module with integrated antennas.

Despite its small size of 9.8 x 17.2 x 1.7 mm, the ISP4520 module integrates the Semtech SX1261-SX1262 chipset which is a LoRa transceiver dedicated to long range IoT applications. This chipset needs an external processor to operate and is coupled with the Nordic Semi nRF52832 SoC. In addition to the powerful ARM Cortex M4F MCU, this chip also provides BLE connectivity for wireless set up and control of the LoRa chip.

The ISP4520 also integrates all passives, crystals and DC/DC converters to optimize the power consumption. And last but not least, it integrates a LoRa-BLE dual band antenna in the same package. This is a new and unique concept developed by Insight SiP within a SIP package, it allows using a proprietary antenna located inside the SiP module.

The ISP4520 has 3 variants:

- ISP4520-EU: Integrates the Semtech SX1261 and the LoRa RF is optimized for 868 MHz.
- ISP4520-US: Integrates the Semtech SX1262 and the LoRa RF is optimized for 915 MHz.
- ISP4520-JP: Integrates the Semtech SX1261 and the LoRa RF is optimized for 923 MHz.

For more details, see Insight SiP module data sheet (document DS4520 module data sheet).

4.2. ISP4520-TB Test Board

The ISP4520-TB Test Board / Mote consists of a module mounted on a PCB for prototyping and testing purposes. It includes a connection to the ISP130603 Interface Board. ISP4520-TB has dimensions of 55 x 75 mm².

It includes:

- ISP4520 LoRa - BLE module
- 4 x footprint for optional 2x5 pin / 1.27mm pitch connector for access to the ISP4520 GPIOs.
- 10 pin FPC connector on top side of the board.
- 14 pin FPC connector on top side of the board.
- 22 pin FPC connector on top side of the board.
- JTAG footprint for programming using 6 pin Segger Jlink Adapter
  https://www.segger.com/products/debug-probes/j-link/accessories/adapters/6-pin-needle-adapter/
- 2 x 5pin header for programming using Segger Jlink interface contained in Nordic Evaluation Board.
- 2-pin header for power supply when using 6 pin JTAG or Nordic JTAG programming options.
- Footprint for optional external LoRa antenna.

The ISP4520C_TB electrical schematic is presented in the document ISP4520C_TB_SCH.

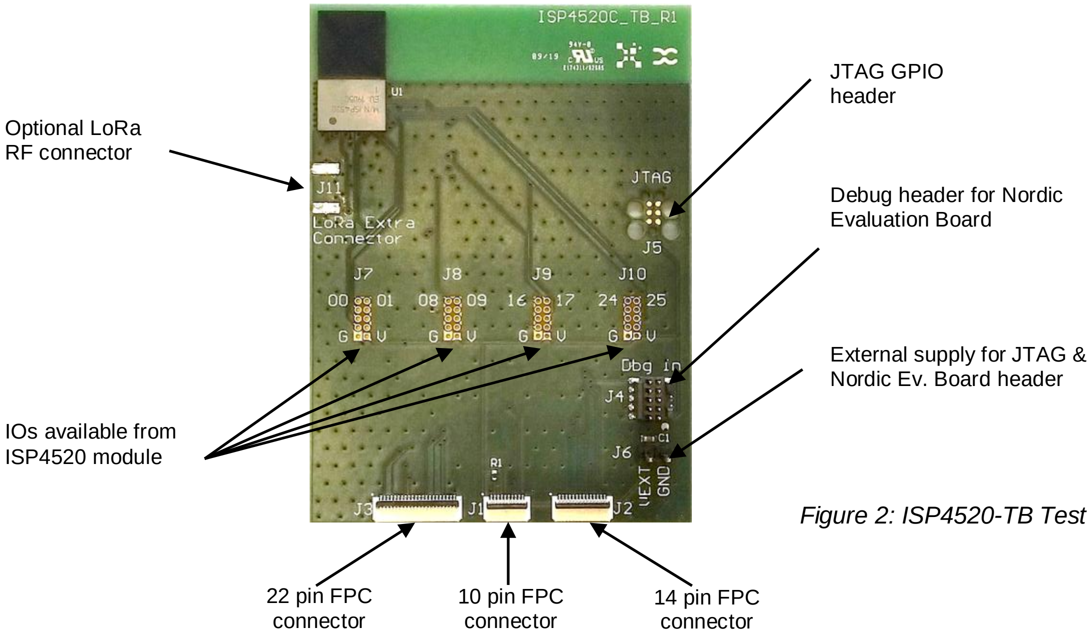

ISP4520C_TB_R1
Optional LoRa RF connector
JTAG GPIO header
Debug header for Nordic Evaluation Board
External supply for JTAG & Nordic Ev. Board header
IOs available from ISP4520 module
22 pin FPC connector
10 pin FPC connector
14 pin FPC connector

Figure 2: ISP4520-TB Test Board

4.3. ISP4520-GW Gateway Board

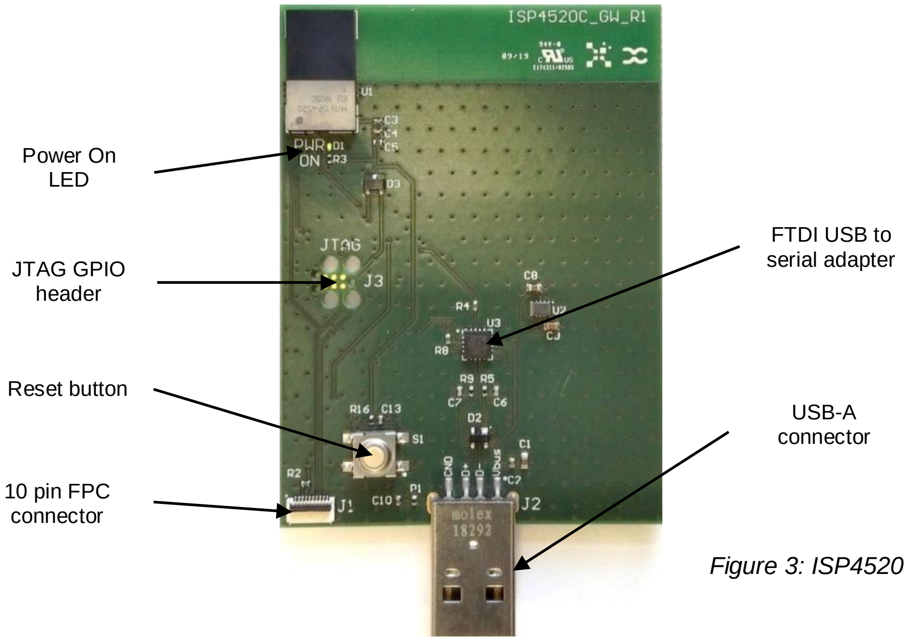


Figure 3: ISP4520-GW Gateway Board

The RX Test Board / Gateway consists of a PCB integrating an ISP4520 module and a USB plug for connection to a PC port com. It enables the communication with a TX Test Board / Mote in a point to point connectivity mode through LoRa standards. ISP4520-GW Gateway Board has dimensions of 55 x 75 mm².

It includes:

- ISP4520 LoRa-BLE module
- 1 x mini-LED for Power ON indication
- USB-A connector
- 10 pin FPC connector for software loading via ISP130603 Interface board
- FTDI USB-to-Serial adapter
- Switch button for ISP4520 reset sequence

The ISP4520C_GW electrical schematic is presented in document ISP4520C_GW_SCH.

4.4. ISP130603 Interface Board

ISP130603 is the application type interface board that has dimensions of 100 x 80 mm². The ISP130603 electrical schematic is presented in document SC130604.

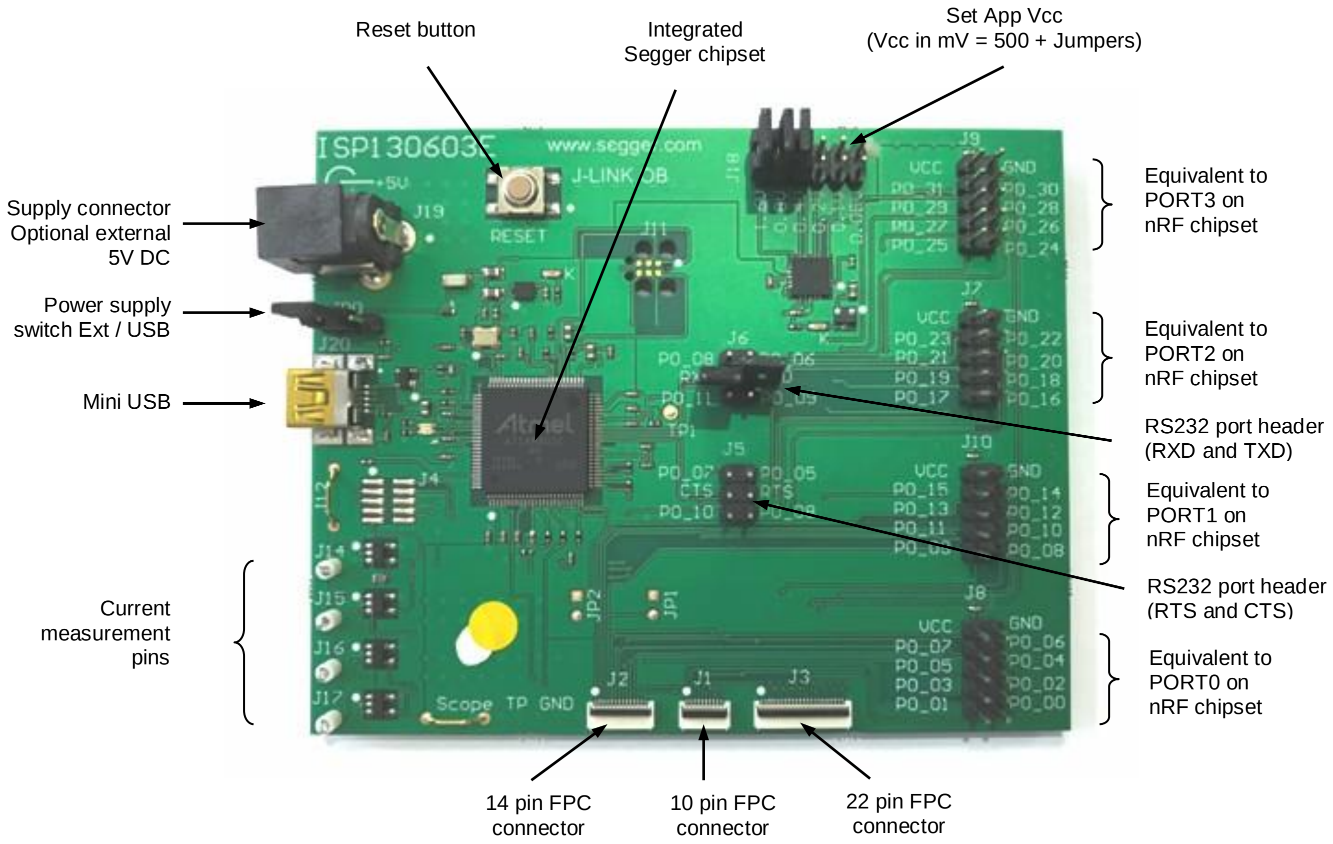

Reset button
Integrated Segger chipset
Set App Vcc (Vcc in mV = 500 + Jumpers)
Supply connector Optional external 5V DC
Power supply switch Ext / USB
Mini USB
Current measurement pins
Equivalent to PORT3 on nRF chipset
Equivalent to PORT2 on nRF chipset
RS232 port header (RXD and TXD)
Equivalent to PORT1 on nRF chipset
RS232 port header (RTS and CTS)
Equivalent to PORT0 on nRF chipset
14 pin FPC connector
10 pin FPC connector
22 pin FPC connector

Figure 4: ISP130603 Interface Board

4.5. ISP4520 Hardware setup

For normal operation, supply voltage is set either by the Interface board or by external supply. It can be set between 2.2V and 3.6V max.

Warning

It is imperative to avoid powering the test boards from more than one power source at a time. This applies to both IS4520-TB and ISP4520-GW. This could lead to permanent damage to the ISP4520 module.

5. LoRa Applications

5.1. LoRa SDK

Insight SiP provides an SDK package containing source code for different LoRa applications. The package will be supplied to ISP4520 module and Dev Kit customers on demand:

- Continuous waveform transmission: Transmit a constant frequency signal, used for debugging purpose.
- Ping-Pong: Exchange simple LoRa packets between 2 devices
- Temperature: Transmit LoRa packets containing temperature data from the ISP4520-TB to the ISP4520-GW
- Class A LoRaWAN: “Hello world” example for Class A LoRaWAN node (LoRaWAN gateway required)
- An example of implementation of LoRaWan AT commands
- These applications are written to be run on ISP4520 (nRF52832 target).

The nRF52 part is based on Nordic SDK v14.2.0 available at *https://developer.nordicsemi.com/nRF5_SDK/nRF5_SDK_v14.x.x*.

The LoRa part is based on Semtech/StackForce implementation from the master branch (4.4.1 release). It is available at: https://github.com/Lora-net/LoRaMac-node/tree/master

The following nRF52832 hardware is used by the LoRa part and therefore shouldn’t be used for other applications:

- RTC2
- SPI0 on pins P0.23, P0.25 and P0.26
- GPIO on pins P0.11, P0.19, P0.24 and P0.27
- GPIOTE on pin P0.11

Detailed LoRa SDK directory tree shown below in Figure 4.

One project is available per application example and for each hardware platform and development environment.

Keil v5 and Segger Embedded Studio are supported.

The Nordic SDK is not included in the archive. Download *nRF5_SDK_14.2.0_17b948a.zip* and extract to <Your directory>\src\nordic_sdk.

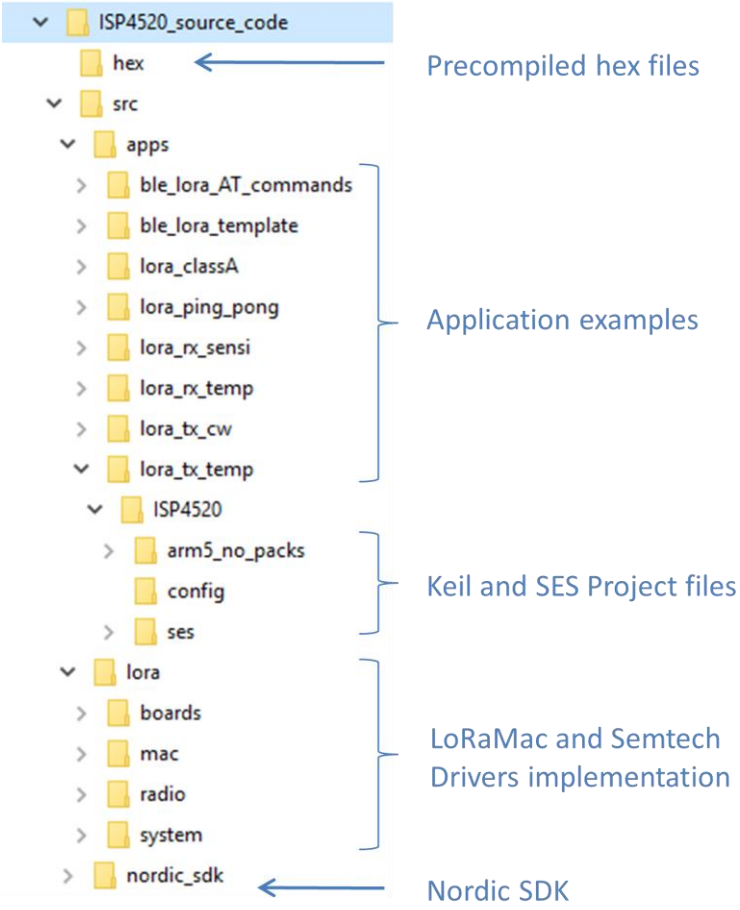


Figure 5: LoRa SDK directory tree

A LoRa RX/TX switch is integrated in the ISP4520. The state of this switch is controlled by the DIO2 pin of the SX126x. The SX126xdriver provides a function called SX126xSetDio2AsRfSwitchCtrl() that allows to do it automatically.

The power of the switch is supplied by the DIO3 pin of the SX126x. This allows to power down the switch when it is not needed and therefore save power. However, there is no function to do it automatically in the SX126x driver. The user needs to implement a specific set of instruction to control the state of the DIO3.

To configure DIO3 as output, the following SX126x register need to be modified:

- Set bit 3 of register 0x0580 to 1
- Set bit 3 of register 0x0583 to 0
- Set bit 3 of register 0x0584 to 0
- Set bit 3 of register 0x0585 to 0
- Set bits [0 to 2] of register 0x0920 to0x06

These configuration steps have to be repeated each time the SX126x exits SLEEP mode.
DIO3 then needs to be set high before any LoRA activity and set low after:

- Set bit 3 of register 0x0920 to 1 (high state) or 0 (low state)

An example of implementation of this control is provided in our LoRa SDK in the SX126x driver specific target board functions implementation file (sx126x-board.c).

5.2. Temperature transmission from ISP4520-TB to ISP4520-GW

Description

In this example, one of the modules is configured as "LoRa temperature transmitter"; it measures the local temperature using its embedded temperature sensor and then transmits the result via LoRa packets to another module configured as "LoRa temperature receiver". The receiver displays the remote temperature on a serial terminal such as RealTerm (*https://sourceforge.net/projects/realterm*)

In the SDK, the "LoRa temperature transmitter" firmware is called "lora_tx_temp" and the "LoRa temperature receiver" is called "lora_rx_temp"

Note: By default, in the development kit, the ISP4520_TB is loaded with the "lora_tx_temp" firmware and the ISP4520_GW is loaded with the "lora_rx_temp" firmware.

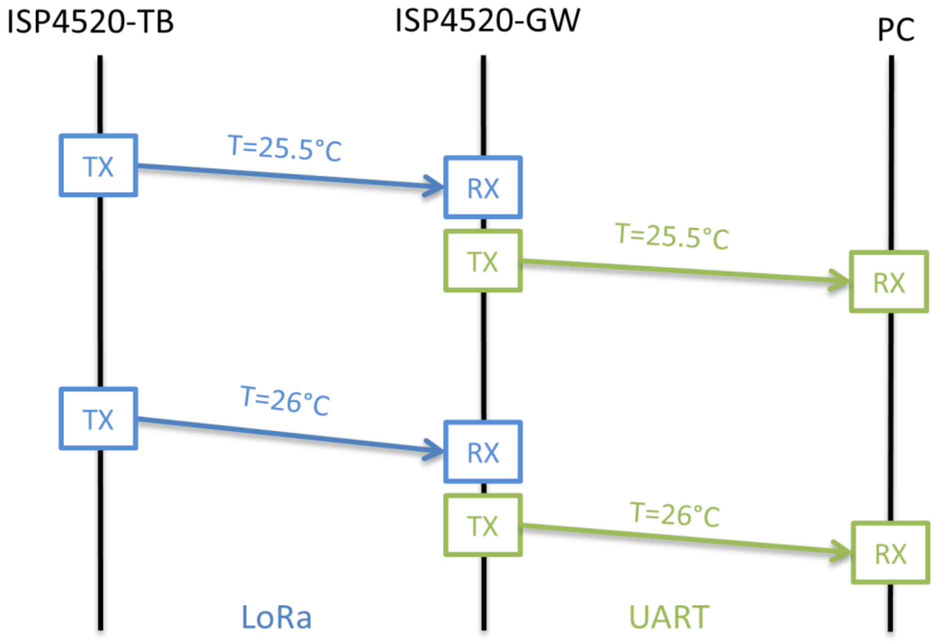


Figure 6: Temperature example

Firmware loading on ISP4520-TB

1. Select one of the methods:

   a. Connect the ISP4520-TB to the ISP130603 Interface Board with the 10 pin FPC jumper cable (0.5 mm pitch, provided in the Development Kit). Connect the provided USB cable from the Interface Board ISP130603 to your computer.

   b. Plug the ISP4520-TB USB to your computer. Connect an external JLINK probe with 6-pin connector (not provided in the kit) to the ISP4520-TB JTAG connector.

   Refer to picture of the hardware configuration below.

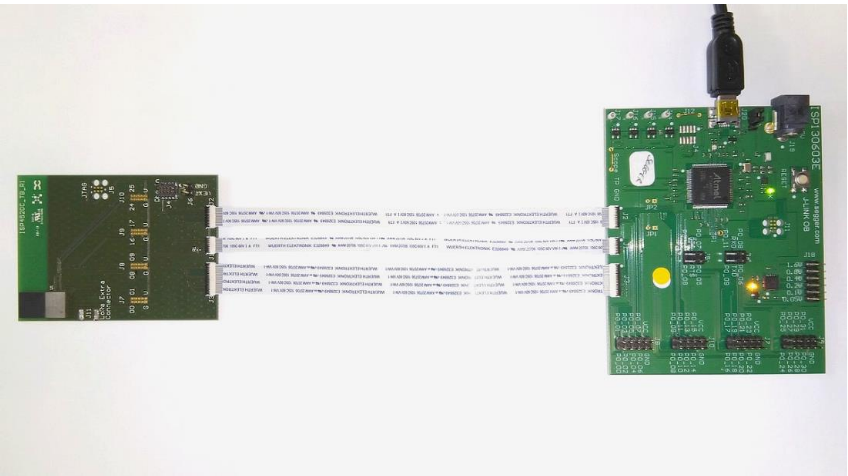


Figure 7: ISP4520-TB firmware loading via embedded JTAG in Interface Board

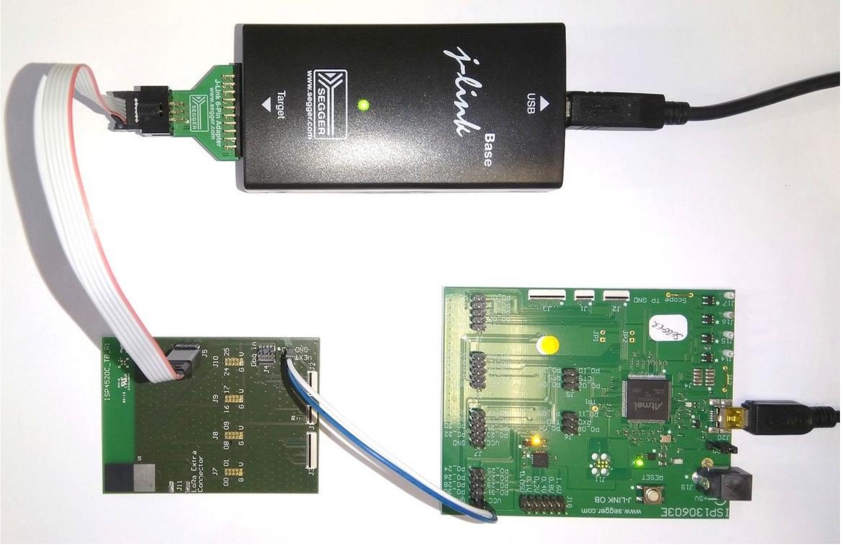
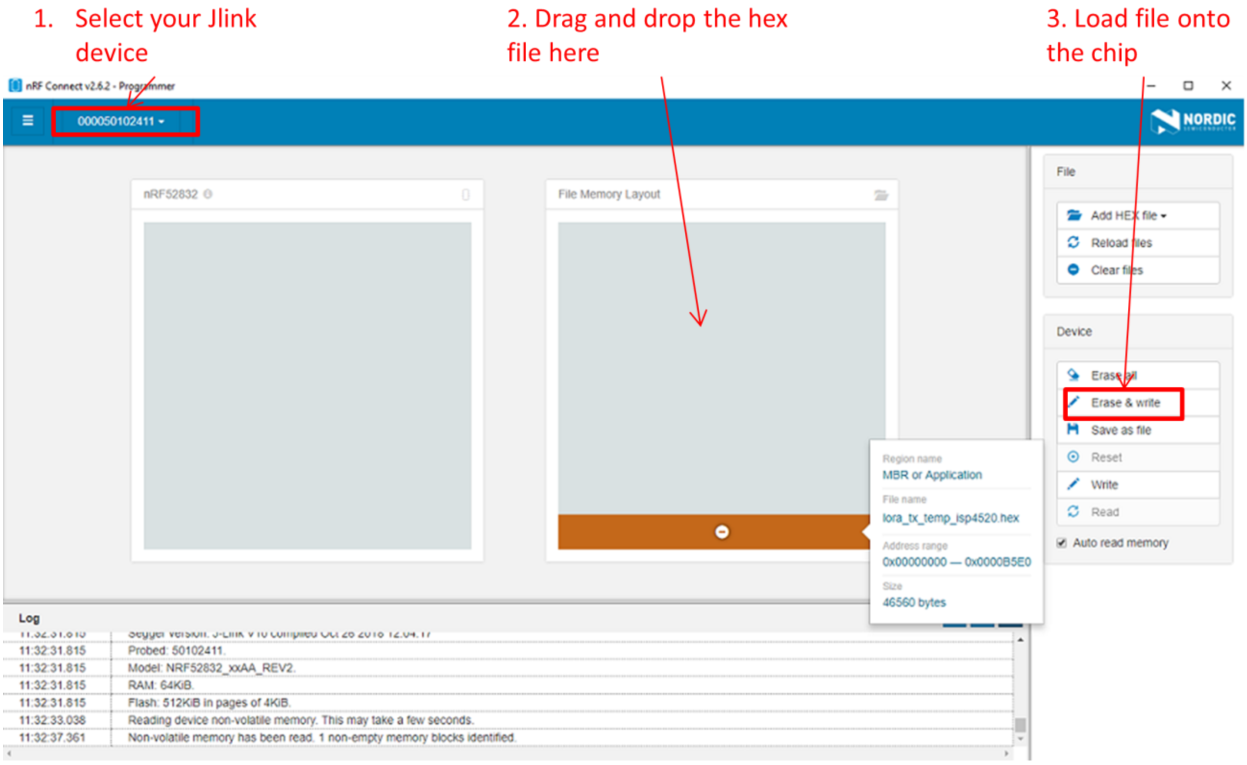


Figure 8: ISP4520-TB firmware loading via JTAG using Segger JLink with 6 pin adapter

2. Start nRF Connect Programmer, drag and drop the binary hex file in the “File Memory Layout” box, and click “Erase & write”. Depending on the module variant the binary file can be rx_temp_ISP4520-EU for Europe, rx_temp_ISP4520-JP for Japan or rx_temp_ISP4520-US for US.



1. Select your Jlink device

2. Drag and drop the hex file here

3. Load file onto the chip

Region name
MBR or Application

File name
lora_tx_temp_isp4520.hex

Address range
0x00000000 — 0x0000B5E0

Size
46560 bytes

Log

```
11:32:31.815   Segger version: J-Link V10 compiled Oct 26 2018 12:04:17
11:32:31.815   Probed: 50102411.
11:32:31.815   Model: NRF52832_xxAA_REV2.
11:32:31.815   RAM: 64KiB.
11:32:31.815   Flash: 512KiB in pages of 4KiB.
11:32:33.038   Reading device non-volatile memory. This may take a few seconds.
11:32:37.361   Non-volatile memory has been read. 1 non-empty memory blocks identified.
```

Figure 9: Loading "lora_tx_temp" with nRF Connect

Firmware loading on ISP4520-GW

1. Select one of the methods:

   a. Connect the ISP4520-GW to the ISP130603 Interface Board with the 10 pin FPC jumper cable (0.5 mm pitch, provided in the Development Kit). Connect the provided USB cable from the Interface Board ISP130603 to your computer.

   b. Plug the ISP4520-GW USB to your computer. Connect an external JLINK probe with 6-pin connector (not provided in the kit) to the ISP4520-GW JTAG connector.

   Refer to picture of the hardware configuration below.

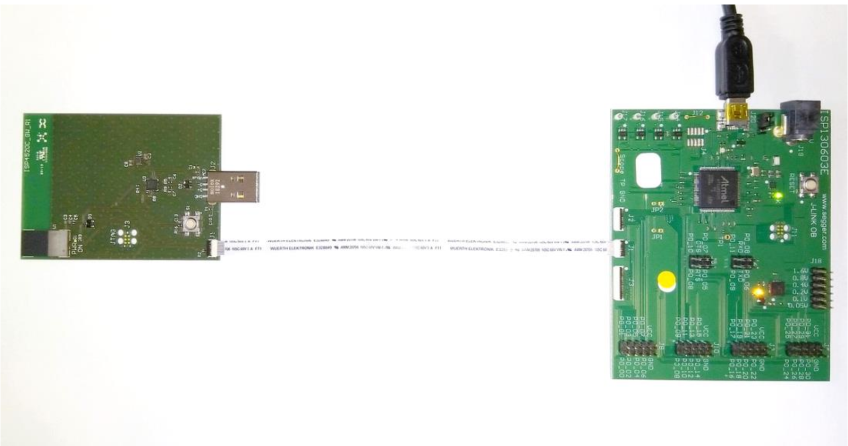


Figure 10: ISP4520-GW firmware loading via embedded JTAG in Interface Board

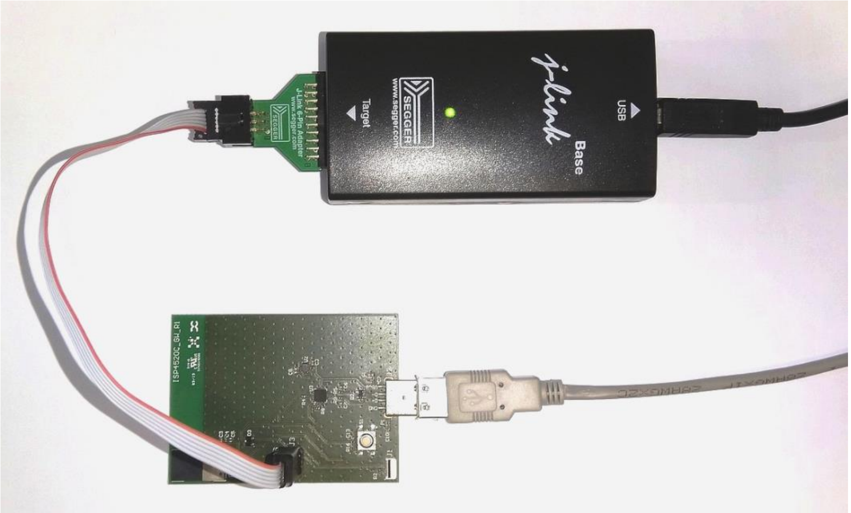
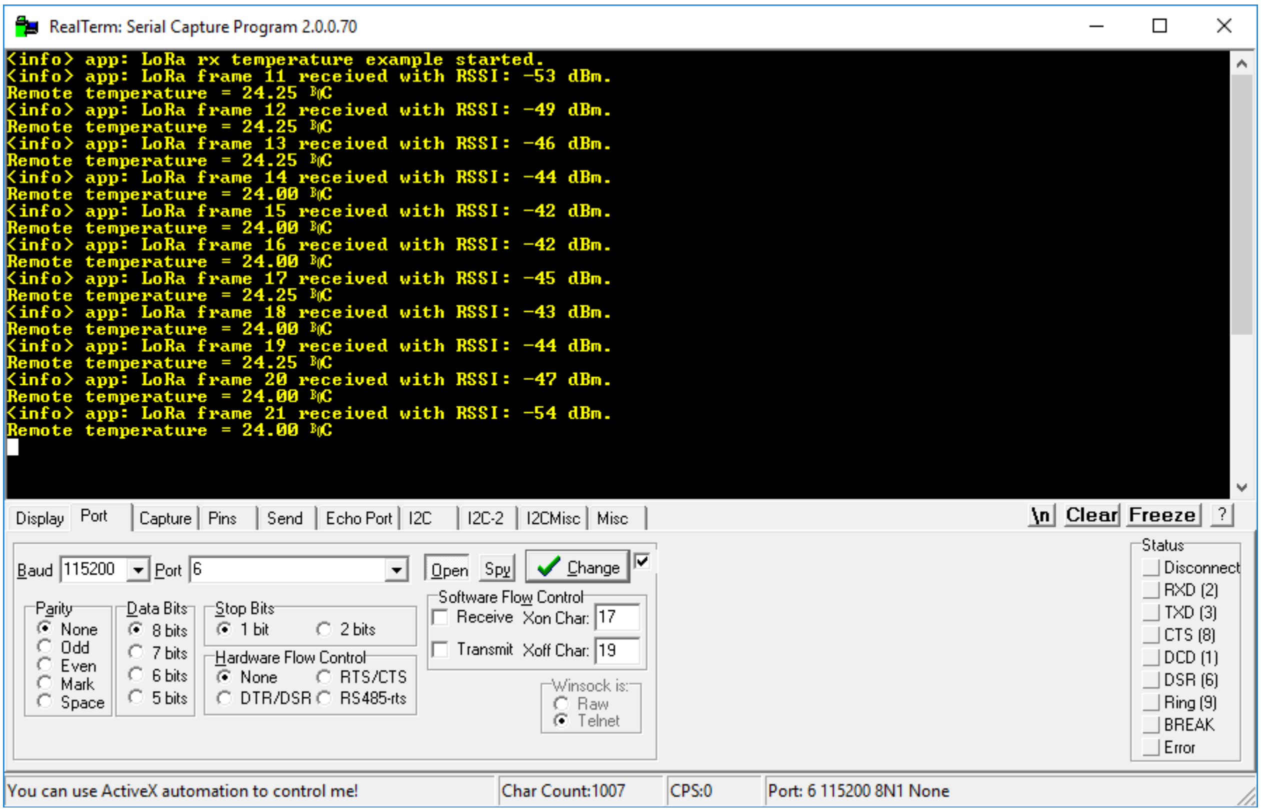


Figure 11: ISP4520-GW firmware loading via JTAG using Segger JLink with 6 pin adapter

2. Start nRF Connect Programmer, drag and drop the binary hex file in the “File Memory Layout” box, and click “Erase & write”. Depending on the module variant the binary file can be tx_temp_ISP4520-EU for Europe, tx_temp_ISP4520-JP for Japan or tx_temp_ISP4520-US for US.

3. If not done already, plug the ISP4520-GW USB to your computer. Download and install a serial terminal software (RealTerm, putty, etc …)

4. Select the port number associated with the ISP4520-GW. Configure the Port with Baudrate:115200, Parity: None, Data bits:8, Stop bits: 1, Hardware Flow control: None



```
<info> app: LoRa rx temperature example started.
<info> app: LoRa frame 11 received with RSSI: -53 dBm.
Remote temperature = 24.25 °C
<info> app: LoRa frame 12 received with RSSI: -49 dBm.
Remote temperature = 24.25 °C
<info> app: LoRa frame 13 received with RSSI: -46 dBm.
Remote temperature = 24.25 °C
<info> app: LoRa frame 14 received with RSSI: -44 dBm.
Remote temperature = 24.00 °C
<info> app: LoRa frame 15 received with RSSI: -42 dBm.
Remote temperature = 24.00 °C
<info> app: LoRa frame 16 received with RSSI: -42 dBm.
Remote temperature = 24.00 °C
<info> app: LoRa frame 17 received with RSSI: -45 dBm.
Remote temperature = 24.25 °C
<info> app: LoRa frame 18 received with RSSI: -43 dBm.
Remote temperature = 24.00 °C
<info> app: LoRa frame 19 received with RSSI: -44 dBm.
Remote temperature = 24.25 °C
<info> app: LoRa frame 20 received with RSSI: -47 dBm.
Remote temperature = 24.00 °C
<info> app: LoRa frame 21 received with RSSI: -54 dBm.
Remote temperature = 24.00 °C
```

Baud 115200 Port 6

Parity: ● None ○ Odd ○ Even ○ Mark ○ Space

Data Bits: ● 8 bits ○ 7 bits ○ 6 bits ○ 5 bits

Stop Bits: ● 1 bit ○ 2 bits

Hardware Flow Control: ● None ○ RTS/CTS ○ DTR/DSR ○ RS485-rts

Software Flow Control: □ Receive Xon Char: 17 □ Transmit Xoff Char: 19

Winsock is: ○ Raw ● Telnet

You can use ActiveX automation to control me! | Char Count:1007 | CPS:0 | Port: 6 115200 8N1 None

Figure 12: Serial terminal with temperature

Note: If the port COM is not found and you get a “USB device not recognized”, try installing the Vitual port COM drivers at *https://www.ftdichip.com/Drivers/VCP.htm*.

LoRa settings

The default LoRa settings used in this example are:

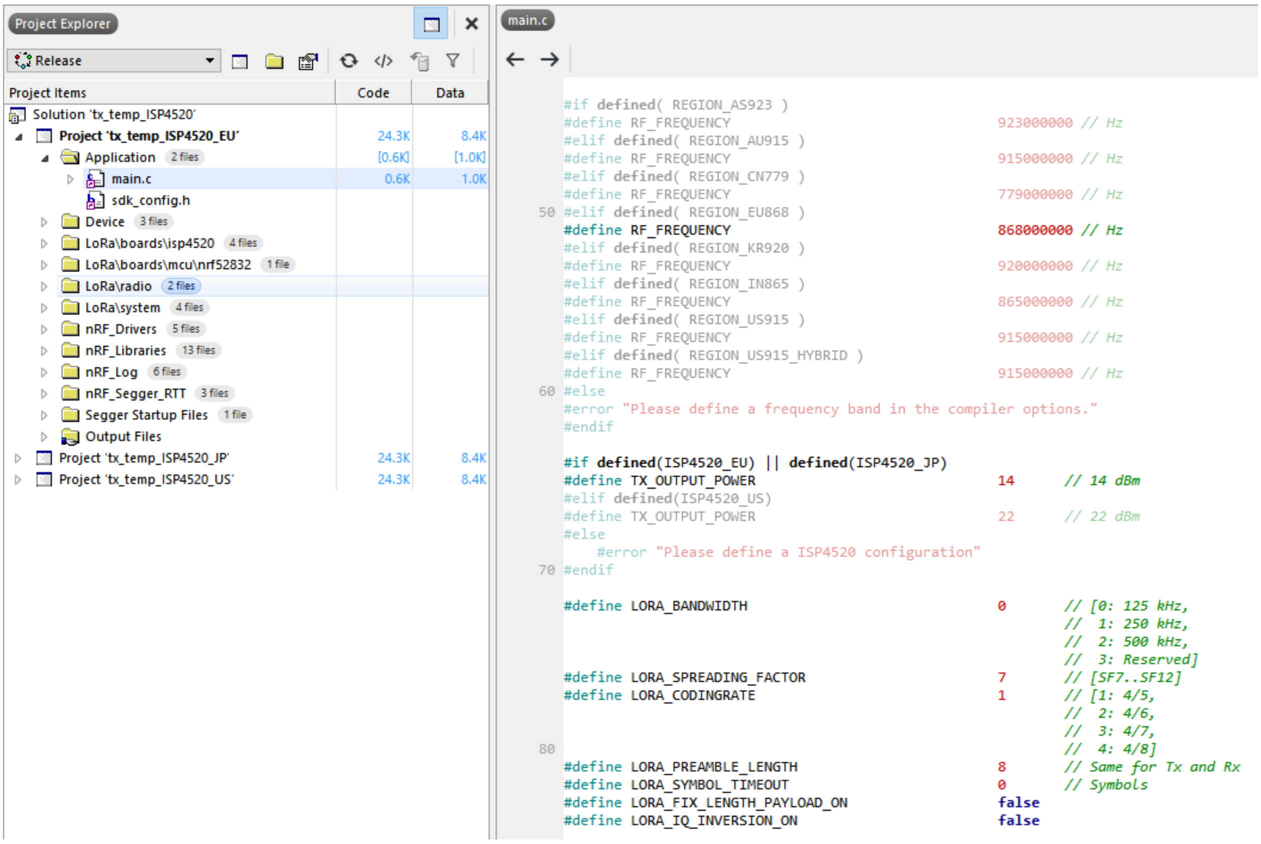
- Bandwidth: 125 kHz
- Spreading factor: 7
- Coding Rate: 4/5
- Frequency: 868 MHz

To change these settings, the source code needs to be modified on both ends.

1. Using Segger Embedded Studio, open the solution named “tx_temp_ISP4520.emProject” in <Your directory>\src\apps\lora_tx_temp\ISP4520\ses.

2. The solution contains 3 projects, one for each region EU, JP and US. Select the project corresponding to the module variant, right click on it and click “Set as active project”.

3. Open main.c file and edit the following #define:
   RF_FREQUENCY
   TX_OUTPUT_POWER
   LORA_BANDWIDTH
   LORA_SPREADING_FACTOR
   LORA_CODINGRATE



```
Project Items                              Code     Data
Solution 'tx_temp_ISP4520'
  Project 'tx_temp_ISP4520_EU'             24.3K    8.4K
    Application  2 files                   [0.6K]   [1.0K]
      main.c                               0.6K     1.0K
      sdk_config.h
    Device  3 files
    LoRa\boards\isp4520  4 files
    LoRa\boards\mcu\nrf52832  1 file
    LoRa\radio  2 files
    LoRa\system  4 files
    nRF_Drivers  5 files
    nRF_Libraries  13 files
    nRF_Log  6 files
    nRF_Segger_RTT  3 files
    Segger Startup Files  1 file
    Output Files
  Project 'tx_temp_ISP4520_JP'             24.3K    8.4K
  Project 'tx_temp_ISP4520_US'             24.3K    8.4K
```

```
#if defined( REGION_AS923 )
#define RF_FREQUENCY                                923000000 // Hz
#elif defined( REGION_AU915 )
#define RF_FREQUENCY                                915000000 // Hz
#elif defined( REGION_CN779 )
#define RF_FREQUENCY                                779000000 // Hz
#elif defined( REGION_EU868 )
#define RF_FREQUENCY                                868000000 // Hz
#elif defined( REGION_KR920 )
#define RF_FREQUENCY                                920000000 // Hz
#elif defined( REGION_IN865 )
#define RF_FREQUENCY                                865000000 // Hz
#elif defined( REGION_US915 )
#define RF_FREQUENCY                                915000000 // Hz
#elif defined( REGION_US915_HYBRID )
#define RF_FREQUENCY                                915000000 // Hz
#else
#error "Please define a frequency band in the compiler options."
#endif

#if defined(ISP4520_EU) || defined(ISP4520_JP)
#define TX_OUTPUT_POWER                             14        // 14 dBm
#elif defined(ISP4520_US)
#define TX_OUTPUT_POWER                             22        // 22 dBm
#else
    #error "Please define a ISP4520 configuration"
#endif

#define LORA_BANDWIDTH                              0         // [0: 125 kHz,
                                                              //  1: 250 kHz,
                                                              //  2: 500 kHz,
                                                              //  3: Reserved]
#define LORA_SPREADING_FACTOR                       7         // [SF7..SF12]
#define LORA_CODINGRATE                             1         // [1: 4/5,
                                                              //  2: 4/6,
                                                              //  3: 4/7,
                                                              //  4: 4/8]
#define LORA_PREAMBLE_LENGTH                        8         // Same for Tx and Rx
#define LORA_SYMBOL_TIMEOUT                         0         // Symbols
#define LORA_FIX_LENGTH_PAYLOAD_ON                  false
#define LORA_IQ_INVERSION_ON                        false
```

Figure 13: LoRa settings in main.c

4. Go to Build -> Build tx_temp_ISP4520_XX and then go to Target -> Download tx_temp_ISP4520_XX (where XX is either EU, US or JP)

5. Repeat with the ISP4520-GW using the lora_rx_temp project. All LoRa parameters (except TX_OUTPUT_POWER) must be the same for both devices.

5.3. Ping-Pong

Description

This example is a direct adaptation of ping-pong example provided with the SX126x drivers. Both devices will exchange LoRa packets in a TX/RX point-to-point link. When one of the devices receives a valid LoRa packet (Ping or Pong), it will notify the user the RSSI and the SNR of the packet and then will transmit the other valid LoRa packet back to the other device.

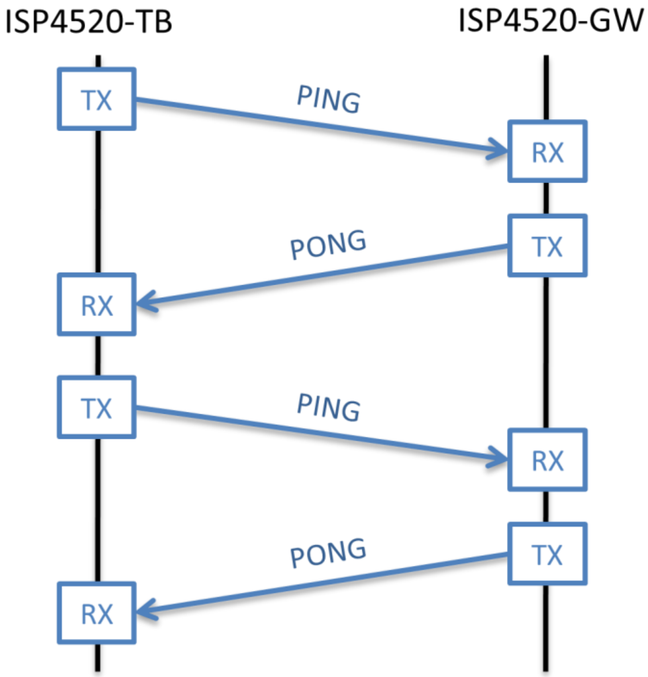
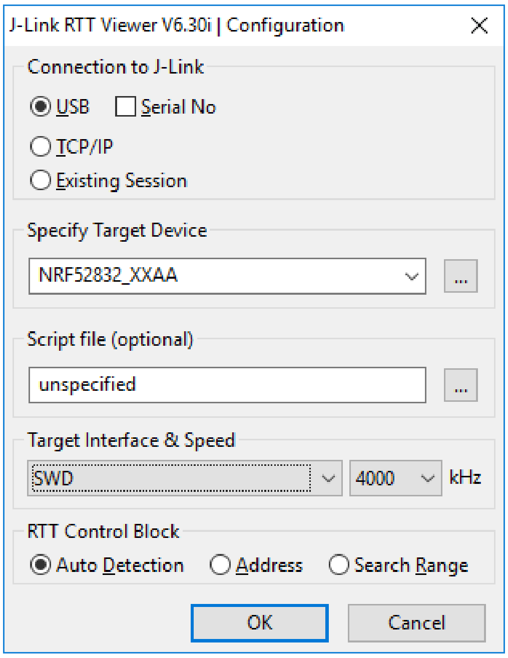


Figure 14: Ping-Pong example

Notifications of received packets are printed on the debug terminal. Open the JLink viewer “JLinkRTTViewer.exe”. By default it is located in “C:\Program Files (x86)\SEGGER\<Your Jlink version>\”.
Specify the target Device: NRF52832_XXA and the interface: SWD.



J-Link RTT Viewer V6.30i | Configuration

Connection to J-Link

● USB □ Serial No

○ TCP/IP

○ Existing Session

Specify Target Device

NRF52832_XXAA

Script file (optional)

unspecified

Target Interface & Speed

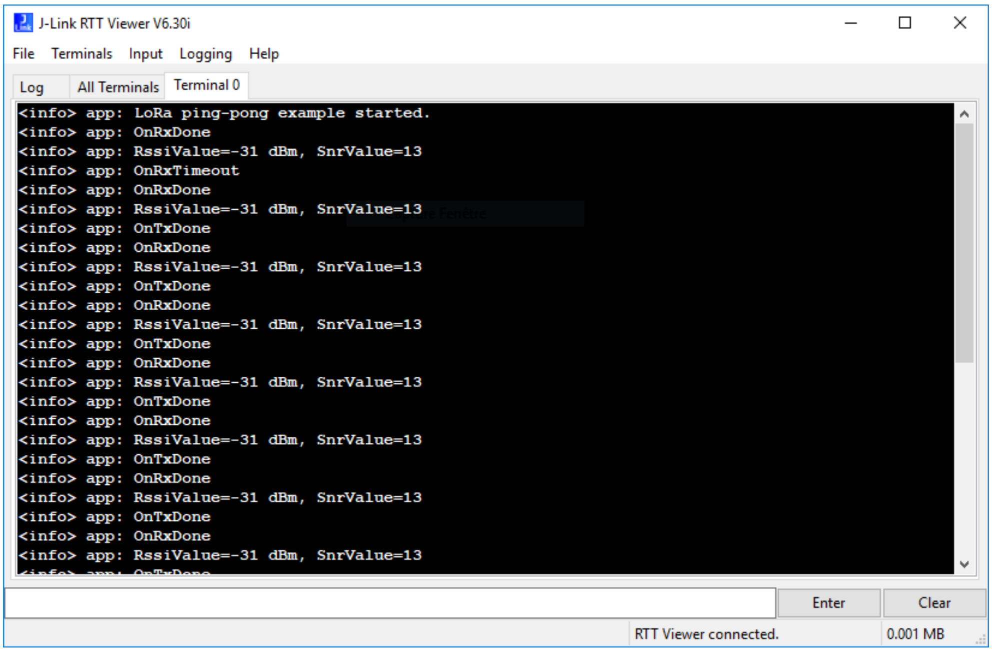
SWD 4000 kHz

RTT Control Block

● Auto Detection ○ Address ○ Search Range

Figure 15: JLink RTT Viewer configuration

A window with the log should appear:



```
<info> app: LoRa ping-pong example started.
<info> app: OnRxDone
<info> app: RssiValue=-31 dBm, SnrValue=13
<info> app: OnRxTimeout
<info> app: OnRxDone
<info> app: RssiValue=-31 dBm, SnrValue=13
<info> app: OnTxDone
<info> app: OnRxDone
<info> app: RssiValue=-31 dBm, SnrValue=13
<info> app: OnTxDone
<info> app: OnRxDone
<info> app: RssiValue=-31 dBm, SnrValue=13
<info> app: OnTxDone
<info> app: OnRxDone
<info> app: RssiValue=-31 dBm, SnrValue=13
<info> app: OnTxDone
<info> app: OnRxDone
<info> app: RssiValue=-31 dBm, SnrValue=13
<info> app: OnTxDone
<info> app: OnRxDone
<info> app: RssiValue=-31 dBm, SnrValue=13
<info> app: OnTxDone
<info> app: OnRxDone
<info> app: RssiValue=-31 dBm, SnrValue=13
<info> app: OnTxDone
```

RTT Viewer connected. 0.001 MB

Figure 16: Ping-Pong logs

Firmware loading

See Section 5.2 above for details about how to load firmware on the ISP4520.

LoRa settings

See Section 5.2 above for details about how to edit the LoRa settings in the firmware.

5.4. Class A LoRaWAN node

Description

This is an example of LoRaWAN end node implementation. It periodically sends a “Hello world!” unconfirmed message to a LoRaWAN network.

The example supports both join procedures: Activation by personalization (ABP) and Over-The-Air-Activation (OTAA). The selection of the join procedure is hardcoded in a configuration file.

LoRaWAN settings

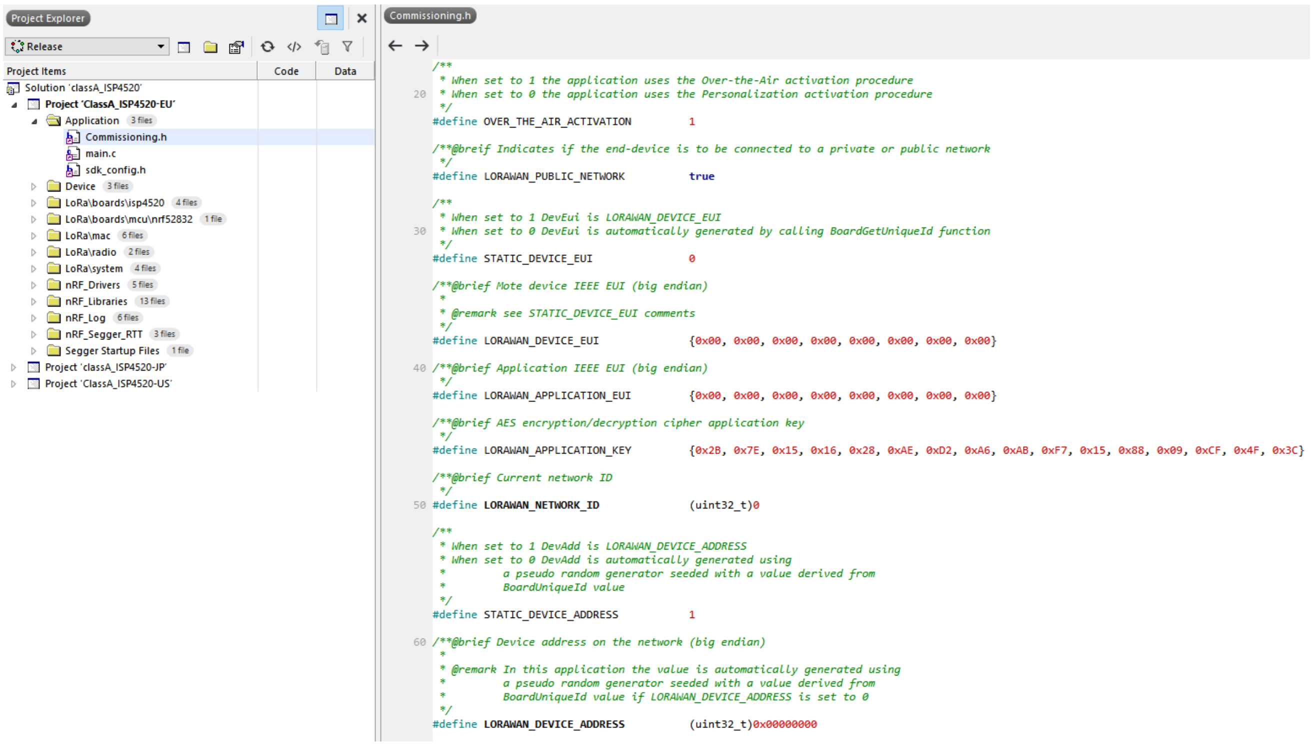
In order to make it work with a LoRaWAN network, the user needs to edit the “commissioning.h” file with all the necessary information such as:

- OTAA or ABP Join procedure
- Device EUI: identifies the device, it can be manually set in the code or automatically generated using nRF52832 device identifier.
- Application EUI: identifies the application.
- Application Key. Used to generate session keys (OTAA only).
- Network Session Key: Encrypt the packets protocol data (ABP only).
- Application Session Key: Encrypt the packet user payload (ABP only).
- Device Address: indentifies a device on a particular network (ABP only). It can be manually set in the code or automatically generating using nRF52832 device identifier.

1. Using Segger Embedded Studio, open the project named “loramac_classA_ISP4520.emProject” in <Your directory>\src\apps\lora_classA\ISP4520\ses.

2. The solution contains 3 projects, one for each region EU, JP and US. Select the project corresponding to the module variant, right click on it and click “Set as active project”.

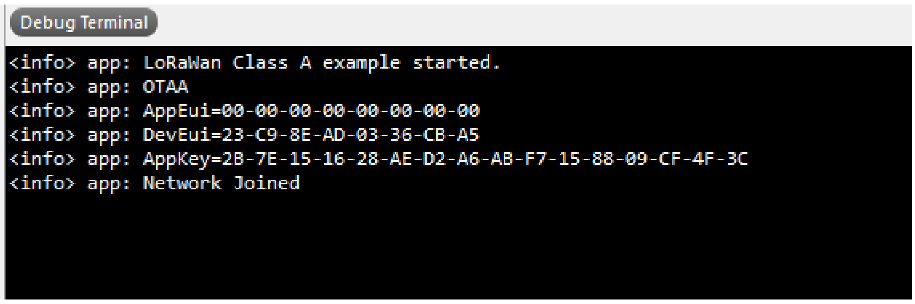
3. Edit the commissioning file to your needs.



```
/**
 * When set to 1 the application uses the Over-the-Air activation procedure
 * When set to 0 the application uses the Personalization activation procedure
 */
#define OVER_THE_AIR_ACTIVATION                 1

/**@breif Indicates if the end-device is to be connected to a private or public network
 */
#define LORAWAN_PUBLIC_NETWORK                  true

/**
 * When set to 1 DevEui is LORAWAN_DEVICE_EUI
 * When set to 0 DevEui is automatically generated by calling BoardGetUniqueId function
 */
#define STATIC_DEVICE_EUI                       0

/**@brief Mote device IEEE EUI (big endian)
 *
 * @remark see STATIC_DEVICE_EUI comments
 */
#define LORAWAN_DEVICE_EUI                      {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00}

/**@brief Application IEEE EUI (big endian)
 */
#define LORAWAN_APPLICATION_EUI                 {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00}

/**@brief AES encryption/decryption cipher application key
 */
#define LORAWAN_APPLICATION_KEY                 {0x2B, 0x7E, 0x15, 0x16, 0x28, 0xAE, 0xD2, 0xA6, 0xAB, 0xF7, 0x15, 0x88, 0x09, 0xCF, 0x4F, 0x3C}

/**@brief Current network ID
 */
#define LORAWAN_NETWORK_ID                      (uint32_t)0

/**
 * When set to 1 DevAdd is LORAWAN_DEVICE_ADDRESS
 * When set to 0 DevAdd is automatically generated using
 *          a pseudo random generator seeded with a value derived from
 *          BoardUniqueId value
 */
#define STATIC_DEVICE_ADDRESS                   1

/**@brief Device address on the network (big endian)
 *
 * @remark In this application the value is automatically generated using
 *          a pseudo random generator seeded with a value derived from
 *          BoardUniqueId value if LORAWAN_DEVICE_ADDRESS is set to 0
 */
#define LORAWAN_DEVICE_ADDRESS                  (uint32_t)0x00000000
```

Figure 17: Commissioning.h file

4. Press "Start execution" (Green arrow at the top right of the IDE). An RTT debug terminal will open and display the Join parameters. If using OTAA method a message will be displayed to confirm that the device successfully joined the network.



```
<info> app: LoRaWan Class A example started.
<info> app: OTAA
<info> app: AppEui=00-00-00-00-00-00-00-00
<info> app: DevEui=23-C9-8E-AD-03-36-CB-A5
<info> app: AppKey=2B-7E-15-16-28-AE-D2-A6-AB-F7-15-88-09-CF-4F-3C
<info> app: Network Joined
```

Figure 18: Network joined using OTAA

6. BLE Application

6.1. Preliminary

This paragraph shows you how to set up and program a simple BLE application. The example shown in this chapter is the proximity application. This application alerts the user when connected devices are too far apart.

For information about other BLE applications please refer to the ISP1507 application note available at https://www.insightsip.com/products/bluetooth-smart-modules/isp1507.

6.2. Firmware loading

1. Connect the provided USB cable from the Interface Board ISP130603 to your computer.

2. Select one of the methods:

   a. Connect the ISP4520-TB to the ISP130603 Interface Board with the 10 pin FPC jumper cable (0.5 mm pitch, provided in the Development Kit). Connect the provided USB cable from the Interface Board ISP130603 to your computer.

   b. Plug the ISP4520-TB USB to your computer. Connect an external JLINK probe with 6-pin connector (not provided in the kit) to the ISP4520-TB JTAG connector.

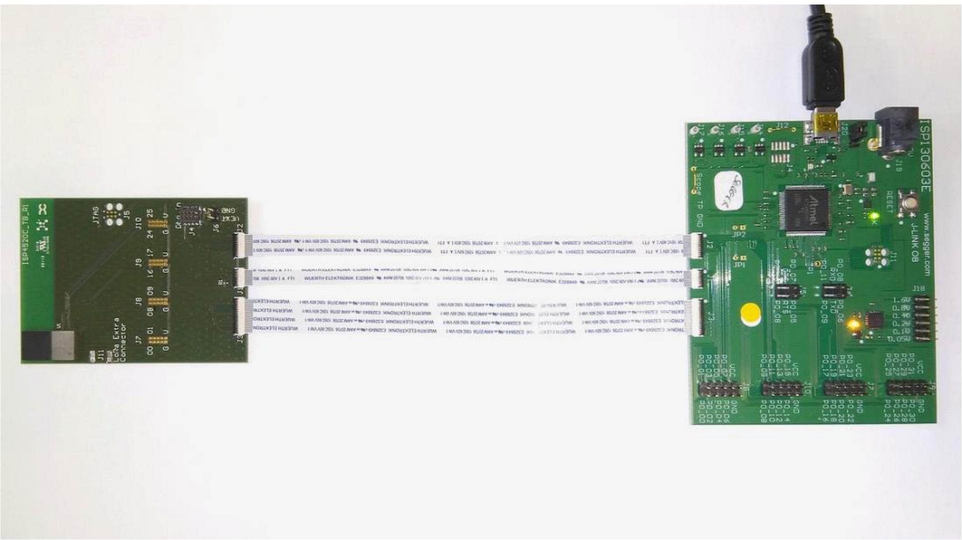


Figure 19: ISP4520-TB firmware loading via embedded JTAG in Interface Board

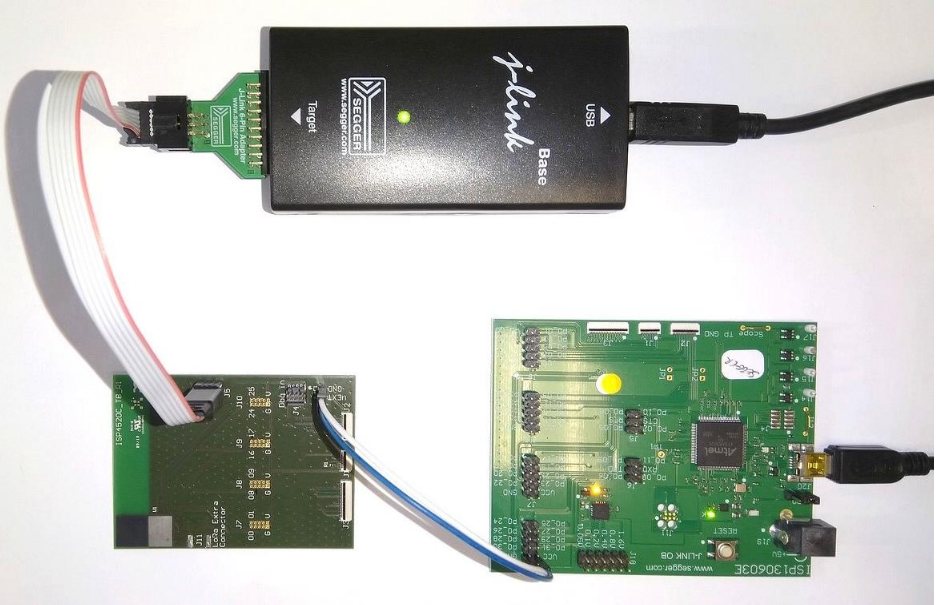
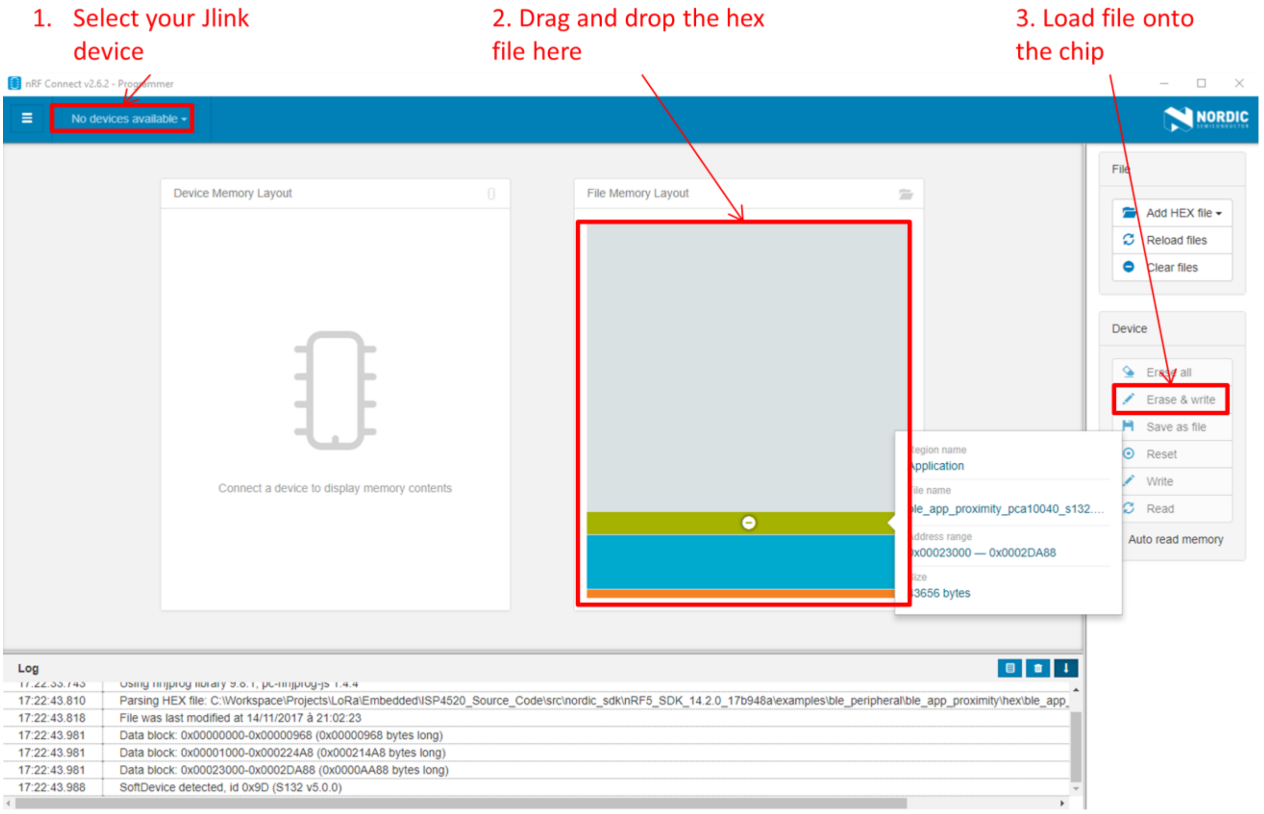


Figure 20: ISP4520-TB firmware loading via JTAG using Segger JLink with 6 pin adapter

Start nRF Connect Programmer, drag and drop the ble_app_proximity_pca10040_s132.hex in the “File Memory Layout” box, and click “Erase & write”. The .hex is located in <your Nordic SDK> \examples\ble_peripheral\ble_app_proximity\hex

Software window shown below.



1. Select your Jlink device

2. Drag and drop the hex file here

3. Load file onto the chip

```
17:22:33.743   Using nrfjprog library 9.8.1, pc-nrfjprog-js 1.4.4
17:22:43.810   Parsing HEX file: C:\Workspace\Projects\LoRa\Embedded\ISP4520_Source_Code\src\nordic_sdk\nRF5_SDK_14.2.0_17b948a\examples\ble_peripheral\ble_app_proximity\hex\ble_app_
17:22:43.818   File was last modified at 14/11/2017 à 21:02:23
17:22:43.981   Data block: 0x00000000-0x00000968 (0x00000968 bytes long)
17:22:43.981   Data block: 0x00001000-0x000224A8 (0x000214A8 bytes long)
17:22:43.981   Data block: 0x00023000-0x0002DA88 (0x0000AA88 bytes long)
17:22:43.988   SoftDevice detected, id 0x9D (S132 v5.0.0)
```

Figure 21: Loading "ble_app_proximity_pca10040_s132" with nRF Connect

6.3. Testing on smartphone

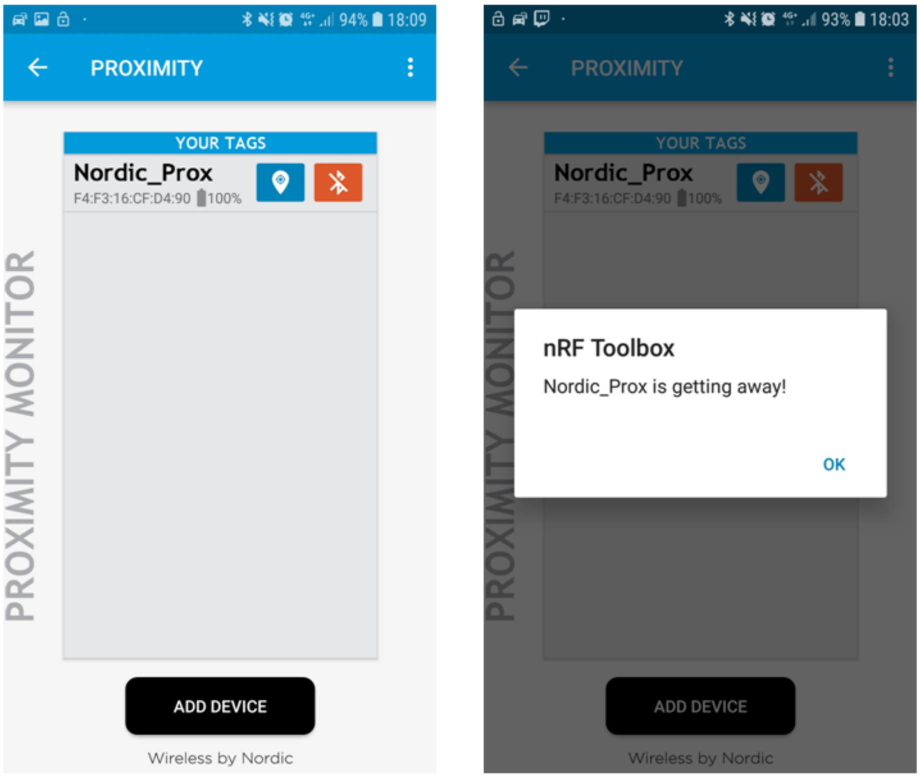
1. On a smartphone download and install the nRF Toolbox application. The application is available on both iOS and Android.

2. Start nRF Toolbox and select PROXIMITY.

3. Click ADD DEVICE to start scanning for suitable BLE devices. The proximity application advertises as “Nordic_prox”. Select the device you want to connect to.

5. Now the user should be notified when both devices are getting far away.

Smartphone window examples shown below.



nRF Toolbox

Nordic_Prox is getting away!

OK

Figure 22: Proximity application