life.augmented

UM2288
User manual

STM32CubeMonitor-RF software tool for wireless performance measurements

Introduction

STM32CubeMonitor-RF (STM32CubeMonRF) is a software tool, which helps designers to test their products based on STMicroelectronics STM32 wireless microcontrollers.

The tool performs the following operations:

- It sends and receives test packets to check the efficiency of radio frequency boards and compute the packet error rate (PER) on Bluetooth® low energy and 802.15.4 technologies.
- It sends commands to Bluetooth low energy controller for standardized tests.
- It sends and receives Bluetooth low energy commands for fast application prototyping.
- It configures a variety of beacons via Bluetooth low energy commands.
- It transfers data over-the-air (OTA) from one device to another, to configure or program a remote device without a wired connection.
- It sends commands to an OpenThread device for application prototyping.
- It explores a Thread® network and displays it with all the relevant information.
- It provides a sniffer tool to analyze 802.15.4 frames with Wireshark.

STM32 CubeMonitor-RF

STM32 Cube

Contents

1 Getting started 9

1.1 Download and setup 9

1.2 Welcome screen 10

1.3 Main screen 11

1.3.1 Menu bar 11

1.3.2 Connection bar 12

1.3.3 Panels 12

1.3.4 Log area 14

2 Connection to target 15

2.1 Use case description and definitions 15

2.2 VCOM / UART connection 16

2.2.1 VCOM connection 16

2.2.2 UART connection 18

2.2.3 VCP device 19

2.3 Opening COM 20

3 Bluetooth low energy mode 22

3.1 Presentation 22

3.1.1 Panels 22

3.2 ACI Commands panel 23

3.2.1 How to send an ACI command 23

3.2.2 Search function 24

3.2.3 Filter usage 24

3.2.4 How to fill parameters. Fixed field / editable field 24

3.2.5 Log functionalities 25

3.3 RF test panel 28

3.3.1 Test mode transmitter (TX) 29

3.3.2 Test mode receiver (RX) 32

3.3.3 PER 35

3.4 Scripts 40

3.4.1 Launching scripts 41

3.4.2 Script recording 43

3.4.3 Scripts modification . . . 43
3.4.4 Script report . . . 44
3.4.5 List of script commands . . . 45
3.5 OTA transfer . . . 48
3.5.1 OTA presentation . . . 48
3.5.2 OTA procedure . . . 49
3.5.3 Use the tool to perform an OTA update . . . 50
3.6 Beacon . . . 53
3.6.1 Beacon presentation . . . 53
3.6.2 Beacon configuration methods: . . . 54
3.6.3 Configuration of the beacon with STM32CubeMonitor-RF . . . 56
3.7 ACI Utilities . . . 62
3.7.1 Remote services discovering . . . 62
3.7.2 Advertising . . . 69

4 OpenThread mode . . . 71
4.1 Presentation . . . 71
4.1.1 Panel . . . 71
4.2 Commands tab . . . 73
4.3 OpenThread scripts tab . . . 75
4.3.1 OpenThread script example . . . 76
4.3.2 List of script commands . . . 76
4.4 Network Explorer tab . . . 77
4.4.1 Controls . . . 77
4.4.2 Display area . . . 79
4.4.3 Infobox . . . 81
4.4.4 Log area . . . 81

5 802.15.4 RF test mode . . . 82
5.1 Presentation . . . 82
5.2 Transmitter test (TX) mode . . . 83
5.2.1 Frame mode . . . 83
5.2.2 Continuous modulated mode . . . 84
5.3 Receiver test (RX) mode . . . 85
5.3.1 Packet error rate (PER) test . . . 86
5.3.2 Link quality assessment (LQI) test . . . 87

5.3.3 Energy detection (ED) test . . . 87
5.3.4 Channel clear assessment (CCA) test . . . 88
5.4 Packet error rate (PER) mode . . . 89
5.4.1 Connecting the additional device to play the role of a packet generator (tester). . . . 90
5.4.2 Configure the parameters of the tester. . . . 91
5.4.3 Configure the parameters of the device under test (DUT). . . . 92
5.4.4 Configure the measurement. . . . 93

6 802.15.4 Sniffer . . . 96
6.1 Presentation . . . 96
6.2 Prerequisite . . . 96
6.2.1 Sniffer device . . . 96
6.2.2 Wireshark . . . 96
6.2.3 Python . . . 96
6.3 Setup verification . . . 97
6.3.1 Sniffer Launch . . . 98
6.3.2 Select interface . . . 98
6.3.3 Configure channel. . . . 99
6.3.4 Start sniffing. . . . 99

Appendix A Beacon configuration format . . . 100

Revision history . . . 101

List of tables

Table 1. OTA loader address table . . . 49
Table 2. Search filtering . . . 51
Table 3. Measurement setting . . . 94
Table 4. Beacon configuration format . . . 100
Table 5. Document revision history . . . 101

List of figures

Figure 1. Welcome screen . . . 10
Figure 2. Main screen . . . 11
Figure 3. Menu bar . . . 11
Figure 4. Connection bar . . . 12
Figure 5. ACI Commands panel . . . 13
Figure 6. Log area . . . 14
Figure 7. Typical connection with a Nucleo board . . . 15
Figure 8. Connection with a remote device . . . 15
Figure 9. Connection with a second device . . . 16
Figure 10. VCOM connection Bluetooth low energy . . . 16
Figure 11. VCOM connection Thread . . . 17
Figure 12. UART connection . . . 18
Figure 13. VCP connection . . . 19
Figure 14. Opening COM . . . 20
Figure 15. Successful COM . . . 20
Figure 16. ACI Commands panel . . . 22
Figure 17. How to send an ACI command . . . 23
Figure 18. Search button . . . 24
Figure 19. Fixed parameter . . . 24
Figure 20. Editable parameter . . . 25
Figure 21. Predefined values . . . 25
Figure 22. Help details . . . 25
Figure 23. Log functionalities . . . 26
Figure 24. More button . . . 26
Figure 25. Message details . . . 27
Figure 26. Purple error messages . . . 27
Figure 27. Gray second board messages . . . 28
Figure 28. Test mode selection . . . 28
Figure 29. Change the test mode . . . 29
Figure 30. Select the test mode . . . 29
Figure 31. Test mode transmitter . . . 30
Figure 32. Transmitting message . . . 31
Figure 33. Transmitted packets count . . . 31
Figure 34. Test mode receiver . . . 32
Figure 35. RSSI measurement . . . 33
Figure 36. RSSI measurement graph . . . 34
Figure 37. RF RSSI measurement big display . . . 35
Figure 38. PER definition . . . 35
Figure 39. PER tester connection . . . 36
Figure 40. PER tester connected . . . 37
Figure 41. PER tester configuration . . . 38
Figure 42. DUT configuration . . . 38
Figure 43. PER test parameters . . . 39
Figure 44. PER and RSSI measurement graph . . . 40
Figure 45. Launching scripts . . . 41
Figure 46. Script execution . . . 42
Figure 47. Script recording buttons . . . 43
Figure 48. Sample script . . . 43

Figure 49. Script report . . . 44
Figure 50. Script verdict . . . 45
Figure 51. Script pause . . . 46
Figure 52. Example . . . 46
Figure 53. Loop simple example . . . 46
Figure 54. Loop second simple example . . . 47
Figure 55. Loop decrement . . . 47
Figure 56. Loop specific increment . . . 47
Figure 57. Nested loop . . . 47
Figure 58. Loop script verdict example . . . 48
Figure 59. Loop script verdict display . . . 48
Figure 60. Script loop error . . . 48
Figure 61. Search procedure . . . 50
Figure 62. Scanning . . . 51
Figure 63. No device found . . . 51
Figure 64. Device found . . . 52
Figure 65. Select the device and parameters . . . 52
Figure 66. Configuring in OTA . . . 53
Figure 67. Progress bar . . . 53
Figure 68. Beacon presentation . . . 54
Figure 69. Beacon usage . . . 54
Figure 70. Online beacon . . . 55
Figure 71. Offline beacon . . . 55
Figure 72. Beacon over-the-air configuration . . . 56
Figure 73. Selecting the beacon mode . . . 56
Figure 74. Common parameters . . . 57
Figure 75. iBeacon parameters . . . 57
Figure 76. Eddystone UID . . . 58
Figure 77. Eddystone URL . . . 59
Figure 78. Online mode transfer configuration . . . 59
Figure 79. Offline mode transfer configuration . . . 59
Figure 80. OTA mode transfer configuration scan . . . 60
Figure 81. OTA mode transfer configuration select device . . . 60
Figure 82. OTA mode transfer configuration transmission . . . 61
Figure 83. ACI Utilities panel . . . 62
Figure 84. Select checkbox . . . 62
Figure 85. Scan parameters . . . 63
Figure 86. Scanning . . . 64
Figure 87. Select device . . . 64
Figure 88. Back . . . 64
Figure 89. Init . . . 64
Figure 90. Connecting . . . 65
Figure 91. Connection error . . . 65
Figure 92. Connected icon . . . 65
Figure 93. Services list . . . 65
Figure 94. Characteristics list . . . 66
Figure 95. Read value . . . 66
Figure 96. Write value . . . 67
Figure 97. Indicate value changed . . . 67
Figure 98. Notify value changed . . . 68
Figure 99. Notifying . . . 68
Figure 100. Advertising parameters . . . 69

Figure 101. Advertising . . . 70
Figure 102. Connected . . . 70
Figure 103. OpenThread - Command tab . . . 71
Figure 104. OpenThread - Script tab . . . 72
Figure 105. OpenThread - Network Explorer tab . . . 72
Figure 106. OpenThread common bottom area . . . 73
Figure 107. OpenThread command-tab - top area . . . 73
Figure 108. Script buttons . . . 74
Figure 109. Command list . . . 74
Figure 110. Command details . . . 74
Figure 111. Scripts tab . . . 75
Figure 112. Sample script . . . 76
Figure 113. OpenThread network explorer tab . . . 77
Figure 114. Auto connection functions . . . 77
Figure 115. Project and background management . . . 78
Figure 116. Explore and size choice controls . . . 78
Figure 117. Display area . . . 79
Figure 118. Zoom and motion controls . . . 80
Figure 119. Infobox . . . 81
Figure 120. Log area . . . 81
Figure 121. Test mode selection . . . 82
Figure 122. Transmitter test mode . . . 83
Figure 123. Field and picklist defining the frame . . . 84
Figure 124. Help frame information . . . 84
Figure 125. Continuous modulated test mode . . . 85
Figure 126. Receiver test mode . . . 85
Figure 127. PER frames received . . . 86
Figure 128. PER frame reception completed . . . 86
Figure 129. LQI measurement . . . 87
Figure 130. ED measurement . . . 88
Figure 131. CCA measurement . . . 89
Figure 132. Packet tester connection . . . 90
Figure 133. PER tester configuration . . . 91
Figure 134. DUT configuration . . . 92
Figure 135. PER test parameters . . . 93
Figure 136. Standard display . . . 94
Figure 137. Chart display . . . 94
Figure 138. Large PER display . . . 95
Figure 139. Large RSSI display . . . 95
Figure 140. Large LQI display . . . 95
Figure 141. Prerequisite check . . . 97
Figure 142. Wireshark interfaces . . . 98
Figure 143. Wheel . . . 98
Figure 144. Channel choice . . . 99
Figure 145. Sniffing . . . 99

1 Getting started

STM32CubeMonitor-RF supports STM32WBxx microcontrollers based on the Arm®(a) Cortex®-M processor.

1.1 Download and setup

STM32CubeMonitor-RF is used with Windows®, Linux®, and Mac® computers.

The information to install the application is described in the release note, which describes the compatibilities and new features available in the tool.

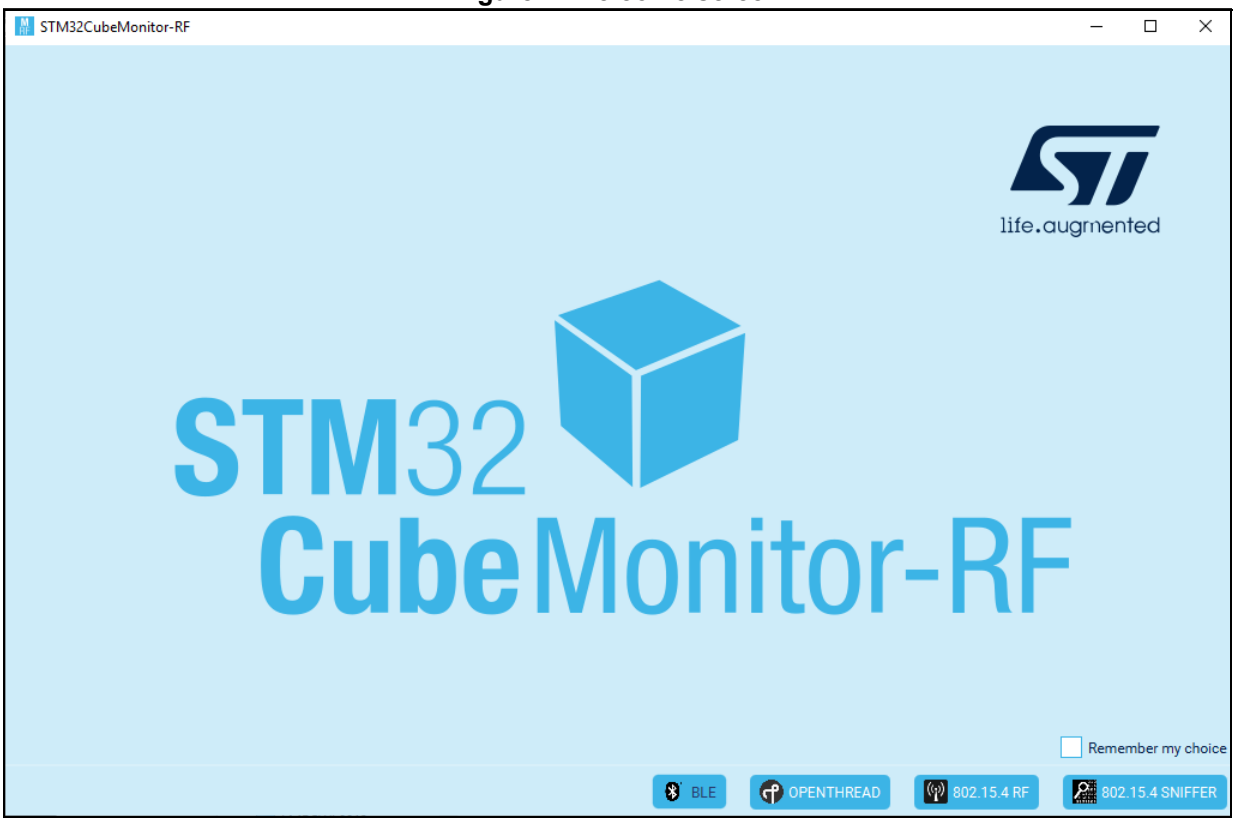
This user manual applies to STM32CubeMonitor-RF version 2.8.0 and later.

Refer to the STM32CubeMonRF release note (RN0104) to install and configure the application.

a. Arm is a registered trademark of Arm Limited (or its subsidiaries) in the US and/or elsewhere.

1.2 Welcome screen

Figure 1. Welcome screen



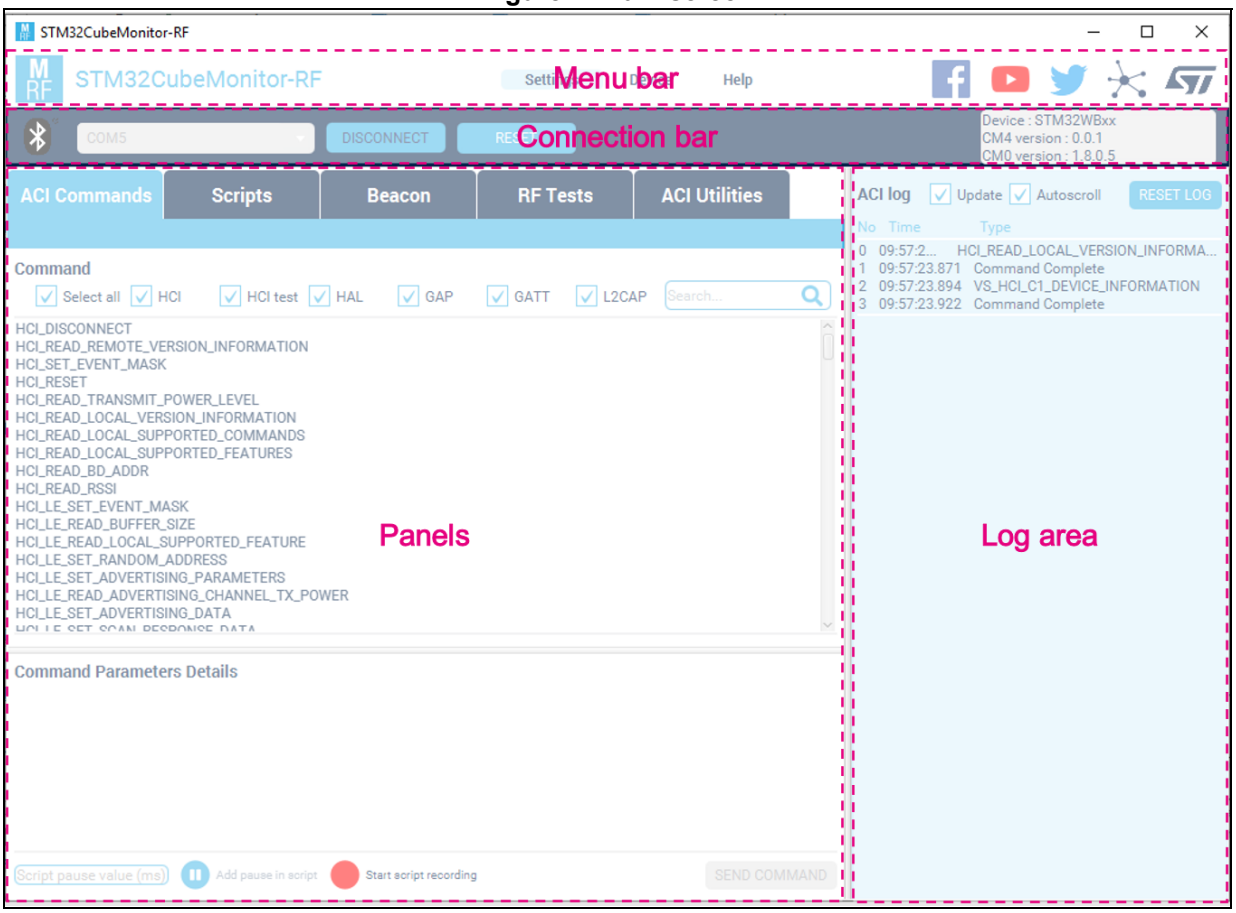
Launching the application opens the welcome screen, where the user selects the mode that he wants to use: Bluetooth low energy, OPENTHREAD, 802.15.4 RF, or 802.15.4 sniffer.

The checkbox *Remember my choice* memorizes the selection, so that the next application launch directly opens it, without the welcome screen, except for 802.15.4 sniffer.

1.3 Main screen

The main screen of the tools is subdivided into four parts: menu bar, connection bar, panels, and log area.

Figure 2. Main screen



1.3.1 Menu bar

Figure 3. Menu bar

The application header provides a menu to use specific tools and display help information.

The *Settings* menu allows the mode change as well as the reset of the default mode choice. The reset of the choice makes the welcome screen appear again.

The *Device* menu provides information and actions related to the connected board.

The *Help* menu provides information about the version of the tool used.

The social network links are available in the right corner. This area contains five shortcuts to access social networks:

- The Facebook™ icon leads to the official STMicroelectronics Facebook page
- The YouTube™ icon leads to the official STMicroelectronics YouTube page
- The Tweeter™ icon leads to the official STMicroelectronics Tweeter page.
- The Share icon leads to the ST Community website
- The STMicroelectronics icon leads to the STMicroelectronics website.

1.3.2 Connection bar

Figure 4. Connection bar

This part displays information related to the device connected to the application.

The icon on the left side reminds the mode selected.

The picklist helps select the COM port to use, buttons allow connecting to, disconnecting from, or resetting the target.

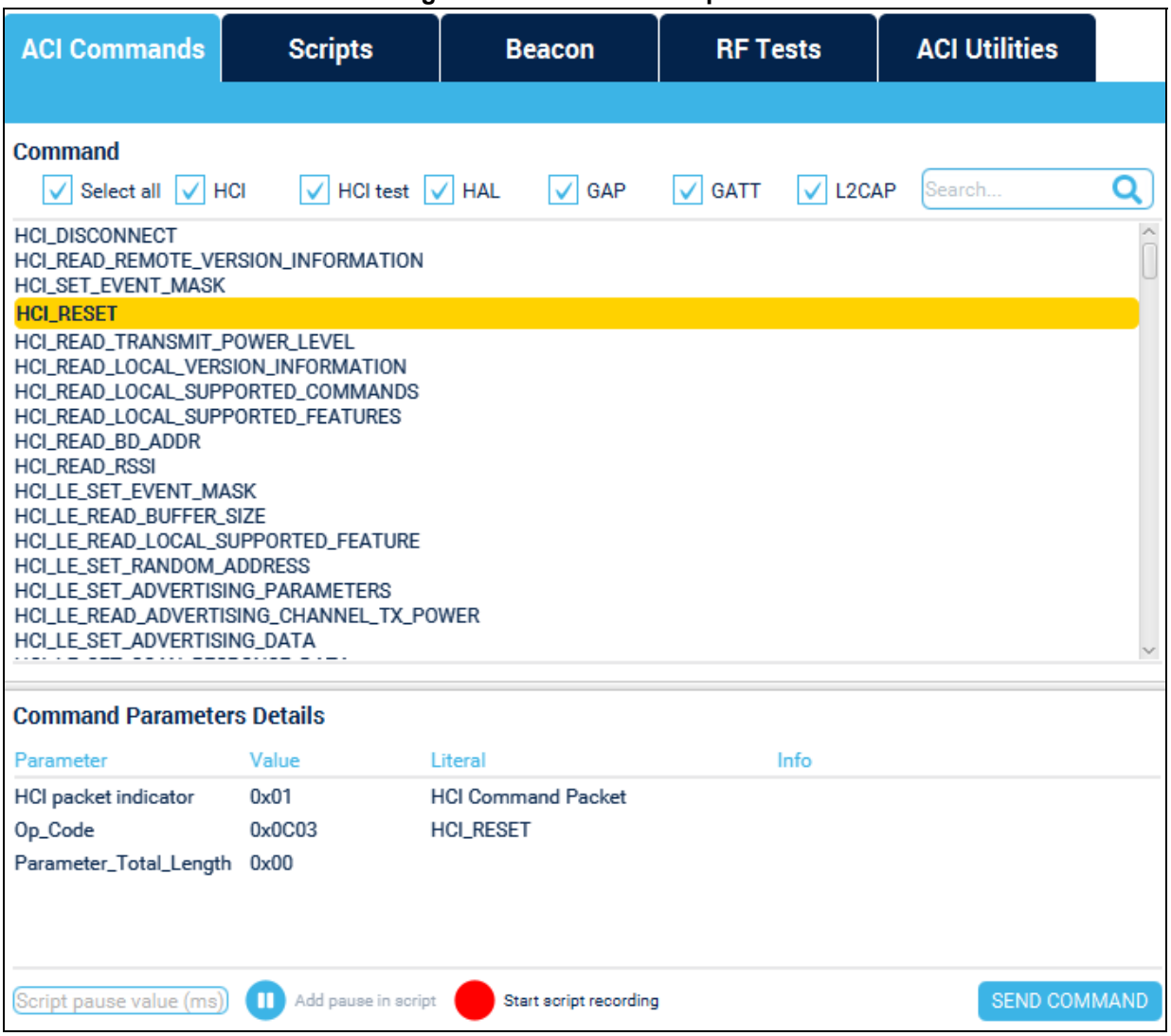
Information about the part connected is displayed on the right.

The *RESET* button is used to reinitialize the Bluetooth low energy wireless stack. When many tests are performed, the button must be used to reset the stack at the start of each test.

1.3.3 Panels

The panels are used to perform specific operations. Each panel regroups different functions. The *ACI Commands* panel example is illustrated in *Figure 5*.

Figure 5. ACI Commands panel

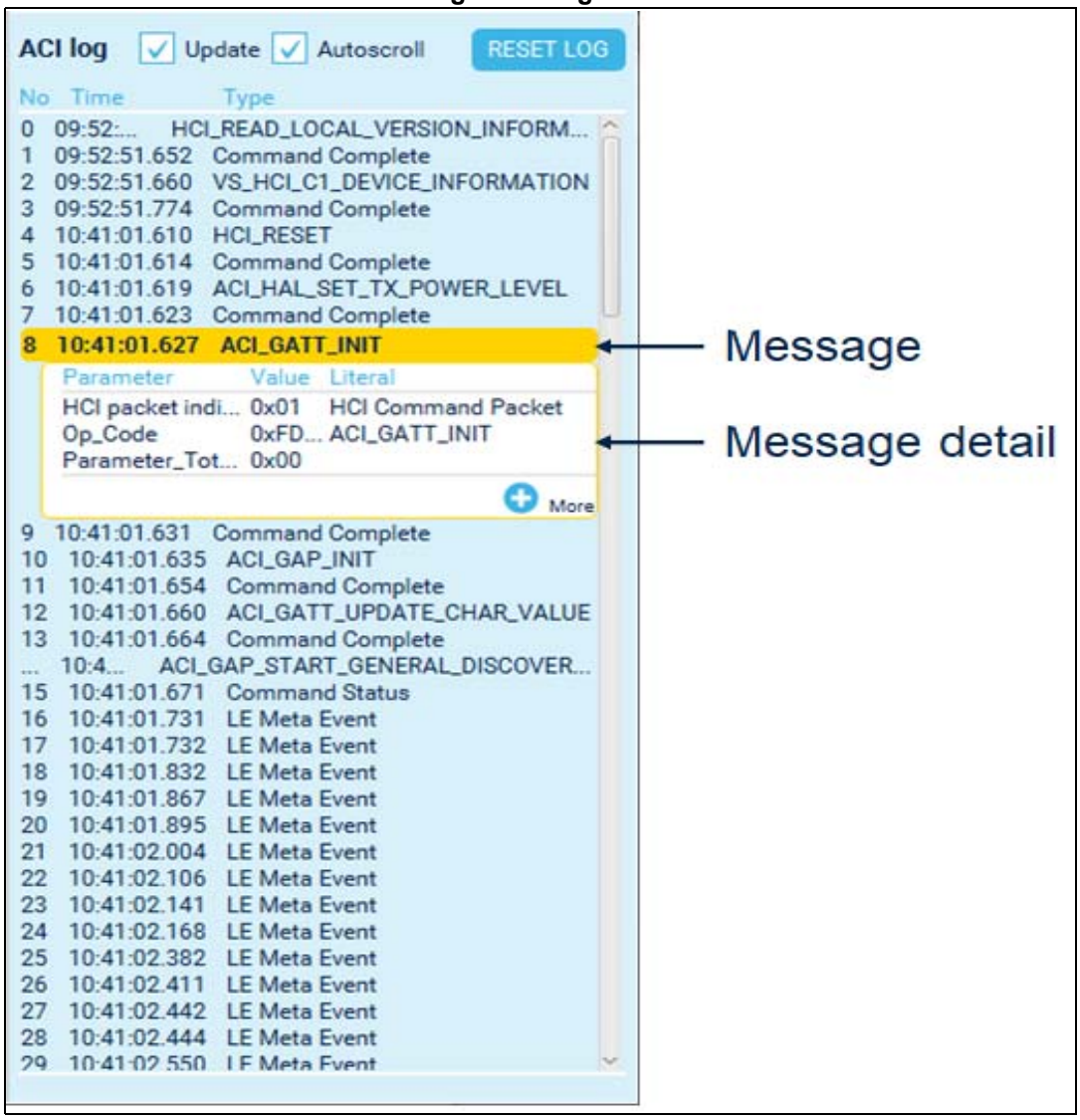


The main panels are *ACI Commands*, *Scripts*, *Beacon*, *RF Tests*, and *ACI Utilities*. Each panel is detailed in a specific section of the document:

- *Section 3.2: ACI Commands panel on page 23*
- *Section 3.4: Scripts on page 40*
- *Section 3.6: Beacon on page 53*
- *Section 3.3: RF test panel on page 28*
- *Section 3.7: ACI Utilities on page 62*

1.3.4 Log area

Figure 6. Log area



The log area shows the messages exchanged between the application and the connected devices. The list shows all message names and details. The log area is described in *Section 3.2.5 on page 25*.

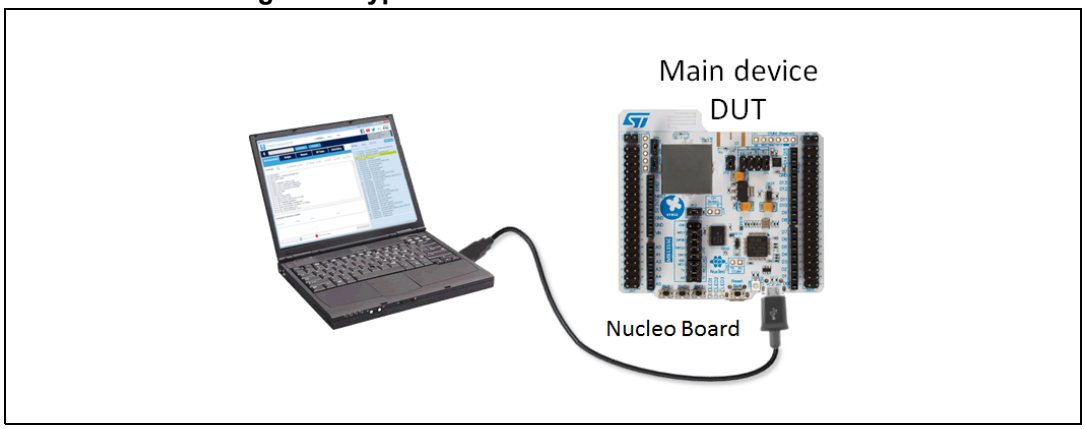
2 Connection to target

2.1 Use case description and definitions

STM32CubeMonitor-RF is usually connected to one STM32WBxx device. The connection is performed through a UART, either by a physical port or a Virtual COM port (VCP).

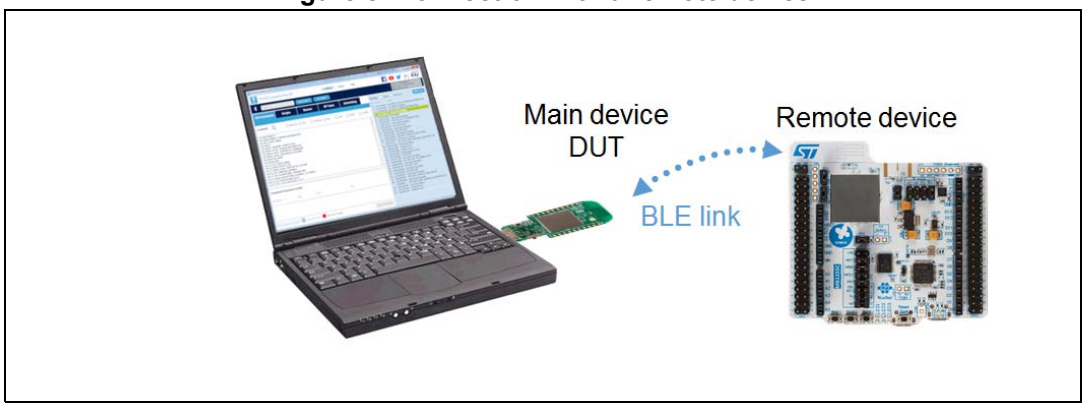
The device connected is usually named the *main device*. This is the board that the user wants to exercise with the tool. It is also named the device under test (DUT).

Figure 7. Typical connection with a Nucleo board



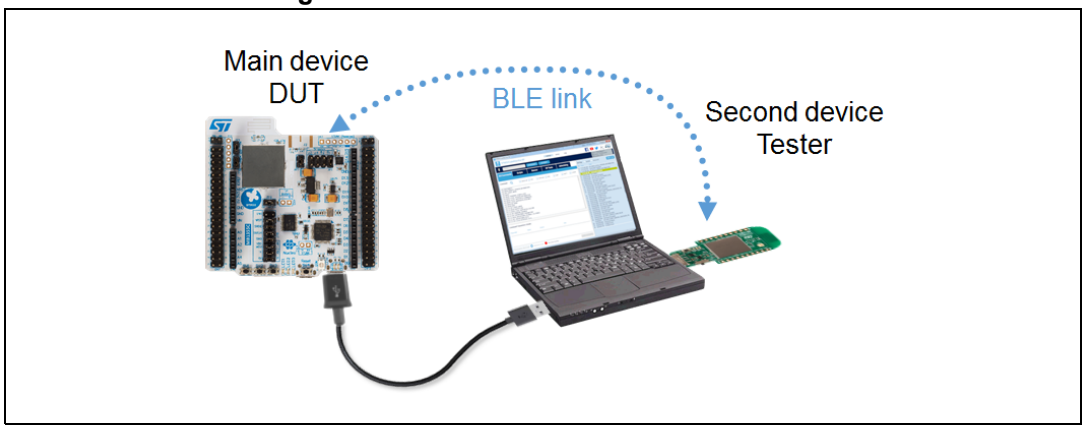
Some operations, like download over-the-air, involve communication with another device. This other device is referred to as the *remote device* in this document.

Figure 8. Connection with a remote device



One RF test makes use of two boards to perform packet transfer error rate measurement. For such a test, a second device is connected; it is named the *second device*. This latter device is the tester, the main device being the device to evaluate (DUT).

Figure 9. Connection with a second device



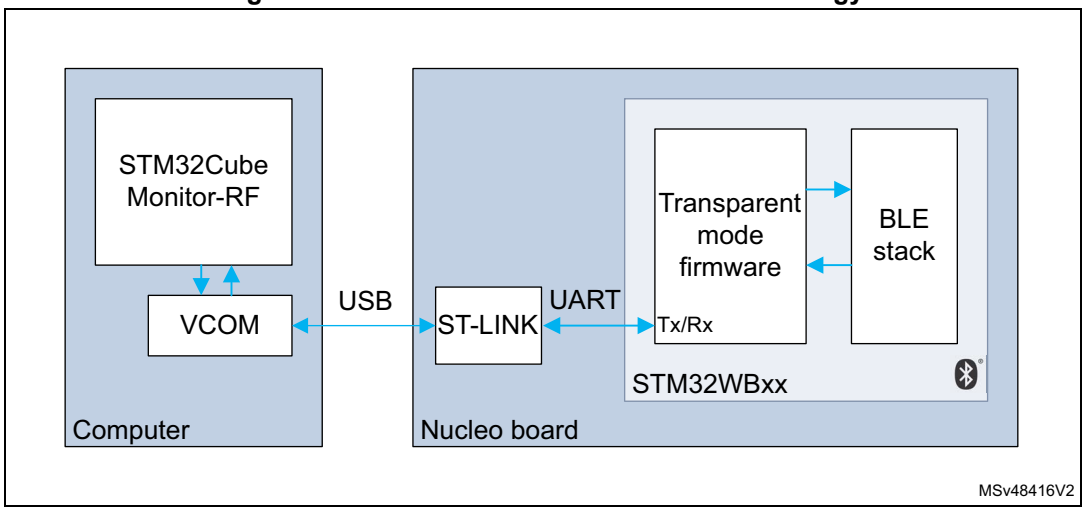
2.2 VCOM / UART connection

The connection must transfer the host controller interface (HCI) or command-line interface (CLI) commands between STM32CubeMonitor-RF and the wireless stack held in the STM32WBxx part. HCI commands are used for Bluetooth low energy applications, CLI commands are used for Thread and 802.15.4 RF tests. The application opens a serial port (virtual or physical) and communicates with the target through this link. Many configurations are possible. The most common ones are described in this section.

2.2.1 VCOM connection

The connection with a Nucleo board uses a Virtual COM port and goes through ST-LINK.

Figure 10. VCOM connection Bluetooth low energy



The application opens the Virtual COM port and sends the data to the VCOM driver.

When a byte is sent, the VCOM transfers the data over USB to the ST-LINK embedded in the Nucleo board. The ST-LINK transfers the data on UART lines to the STM32WBxx controller.

For Bluetooth low energy, a special firmware in STM32WBxx called *transparent mode* copies the data received on the Rx pin to the Bluetooth low energy stack. Data sent back by the Bluetooth low energy stack follows the reverse path.

The transparent mode firmware is available in the STM32CubeWB Firmware Package (Refer to folder *\Projects\xxx\Applications\BLE\BLE_TransparentMode*).

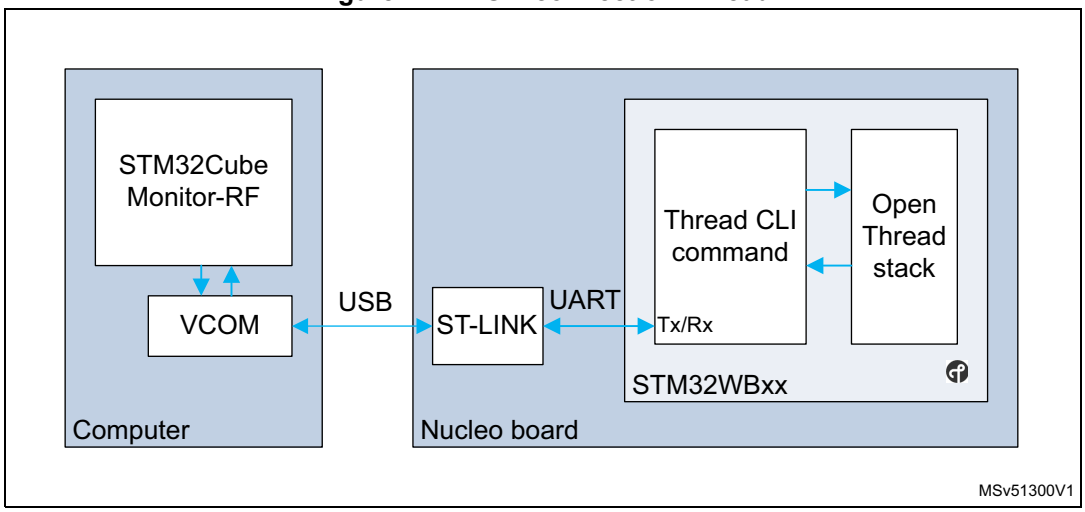
The wireless stack firmware *stm32wb5x_BLE_Stack_full_fw.bin* is available in *\Projects\STM32WB_Copro_Wireless_Binaries*.

For Thread, the *Thread_Cli_cmd* firmware copies the data from the UART to the OpenThread command-line interpreter. Data sent back by the interpreter are forwarded to the UART.

The CLI firmware source code is available in the STM32CubeWB Firmware Package (Refer to folder *\Projects\xxx\Applications\Thread\Thread_Cli_Cmd*).

The wireless stack firmware *stm32wbxx_Thread_FTD_fw.bin* is available in *\Projects\STM32WB_Copro_Wireless_Binaries*.

Figure 11. VCOM connection Thread



For 802.15.4 RF tests, the Cli_Phy_802_15_4 firmware transfers the data from the UART to the 802.15.4 wireless stack. Data sent back by the stack follows the reverse path.

For 802.15.4 RF tests, the Phy_802_15_4_Cli source is available in the STM32CubeWB Firmware Package (Refer to the folder *\Projects\P-NUCLEO-WB55.Nucleo\Applications\Phy_802_15_4\Phy_802_15_4_Cli*).

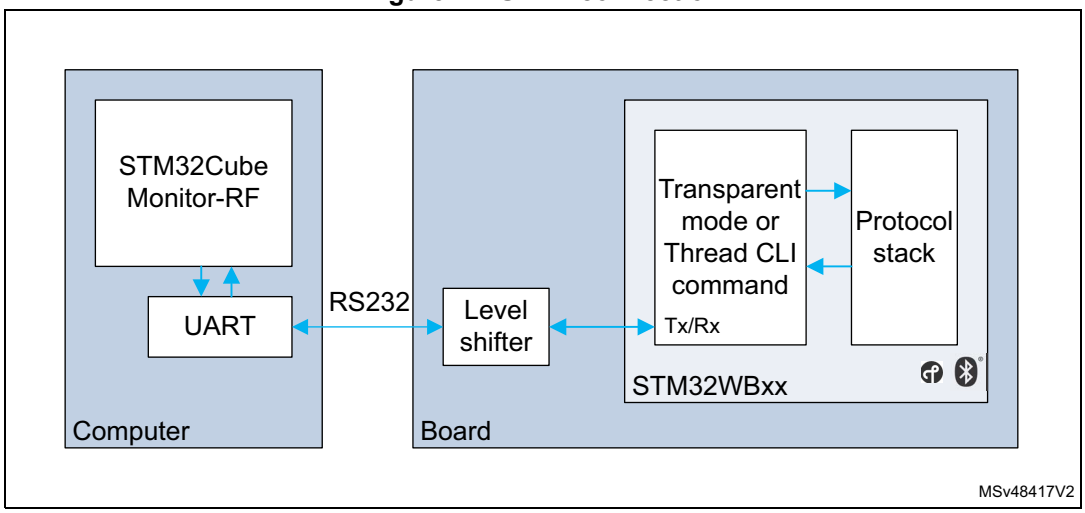
The wireless stack firmware *stm32wb5x_Phy_802_15_4_fw.bin* is available in *\Projects\STM32WB_Copro_Wireless_Binaries*

When the ST-LINK part is replaced by a USB to serial converter, the VCOM driver may be installed automatically on the computer. For the converter without an automatic driver setup, the user must install the VCOM driver manually.

2.2.2 UART connection

It is possible to use a physical UART link to connect directly to any board.

Figure 12. UART connection



In this case, data are sent directly in serial mode through the level shifter. Refer to the transparent mode or CLI command release note for UART configuration.

The UART connection can be used to connect an STM32WB55 USB dongle for 802.15.4 RF tests.

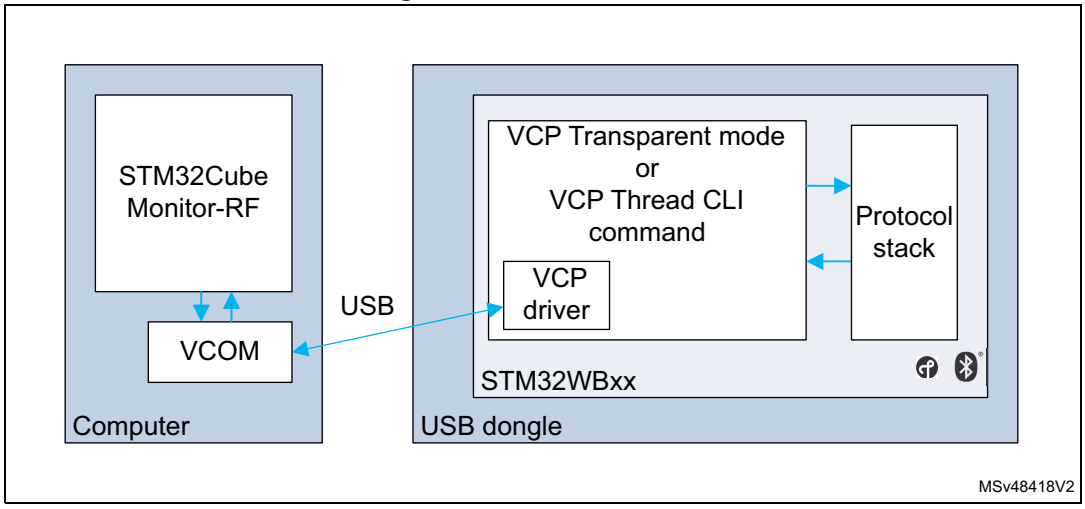
To configure the USB dongle for the 802.15.4 test:

1. Build the Nucleo firmware *Phy_802_15_4_Cli* and flash it. (*STM32Cube_FW_WB_V1.12.0\Projects\P-NUCLEO-WB55.Nucleo\Applications\Phy_802_15_4\Phy_802_15_4_Cli*)
2. Flash with DFU the wireless stack *stm32wb5x_Phy_802_15_4_fw.bin* in the dongle. (Binary in *Projects\STM32WB_Copro_Wireless_Binaries*)
3. Move solder bridge SB2 to SB6 (connection of PB7 to CN2.7).
4. Connect the serial cable to PB7 (PC Tx) and PB6 (PC Rx) (PB7 is CN2.7 and PB6 on CN2.6).

2.2.3 VCP device

In this case, no UART is involved. The data goes directly from the computer to the microcontroller through the USB.

Figure 13. VCP connection



A special VCP firmware is used. It implements a VCP driver to copy the data from the USB port to the protocol stack. The VCOM driver may be installed automatically on the computer or needs to be installed manually by the user. This configuration is used for the STM32WB55 USB dongle reference board and the Nucleo sniffer configuration.

1. For Bluetooth low energy:
   The firmware is in
   \Projects\NUCLEO WB55.USBDongle\Applications\BLE\BLE_TransparentModeVCP.
   The wireless stack is in
   \Projects\STM32WB_Copro_Wireless_Binaries\STM32WB5x\stm32wb5x_BLE_Stack_full_fw.bin
2. For Thread:
   The firmware source code is in
   \Projects\NUCLEO WB55.USBDongle\Applications\Thread\Thread_Cli_Cmd.
   The wireless stack is in
   \Projects\STM32WB_Copro_Wireless_Binaries\STM32WB5x\stm32wb5x_Thread_FTD_fw.bin
3. For 802.15.4 sniffer:
   The wireless stack is in
   \Projects\STM32WB_Copro_Wireless_Binaries\STM32WB5x\stm32wb5x_Mac_802_15_4_fw.bin. The firmware is *Mac_802_15_4_Sniffer.bin*. The location changes with the operating system:
   a) For Windows, the firmware is in the
   <Public documents>\STMicroelectronics\STM32CubeMonitor-RF\sniffer folder,

which means *C:\Users\Public\Documents\STMicroelectronics\STM32CubeMonitor-RF\firmwares*.

b) For Linux, the firmware is in the *<userhome>/STMicroelectronics/STM32CubeMonitor-RF/sniffer folder*.

c) For macOS®, the firmware is inside the document folder provided in the setup package: *\Users\Public\Documents\STMicroelectronics\STM32CubeMonitor-RF\firmwares\Mac_802_15_4_Sniffer.bin*

2.3 Opening COM

The first step to use the application is to connect to the device under test in the connection bar:

Figure 14. Opening COM

The procedure is:

- Connect the board to the computer. If VCOM or VCP is used, a driver needs to be installed; it may take a few seconds at the first connection. For some devices, drivers need to be installed manually.
- Select the serial port to use in the picklist (Comx on Windows and ttyACMx on Linux and Mac).
- Click *CONNECT*

The board is connected, and the version is displayed on the right side of the bar.

Figure 15. Successful COM

When the *CONNECT* button is pressed, the software attempts to communicate with the device to read the firmware and hardware versions. If the connection is not working, the tool displays an error and disconnects the COM port.

Caution: In case of a connection error, the user must check these points:

- When a board is connected for the first time, it takes some time to load the drivers, or the driver may not install automatically. If the tool is not showing the COM port in the list, check that drivers are properly installed.
- Delay on Ubuntu®[a]:
  - On Ubuntu, the modemmanager process is checking the COM port when the board is plugged. Due to this activity, the COM port is busy for a few seconds, and STM32CubeMonitor-RF is unable to connect.
  - The user must wait for the end of the modemmanager activity before opening the COM port.
  - If the modemmanager is not required by the user, it is possible to uninstall it with the command *sudo apt-get purge modemmanager*.
- Port not visible on Linux:
  - The user may not have the proper access rights for ttyACM. In Ubuntu, it is required to add the user to the dial-out group with the command *sudo adduser <username> dialout* (replace username with user name).
- If the port is opened by another application, the tool is unable to connect.
- When a USB device is removed, the Virtual COM port is not closed automatically, and software may not be informed of the disconnection. If a USB device is inserted when the virtual port is already opened, the board is not mounted in the system. To solve this, close the COM port on STM32CubeMonitor-RF, disconnect and re-insert the USB cable. In some rare cases, it is mandatory to enable or disable the COM port in the OS device manager.

a. Ubuntu is a registered trademark of Canonical Ltd.

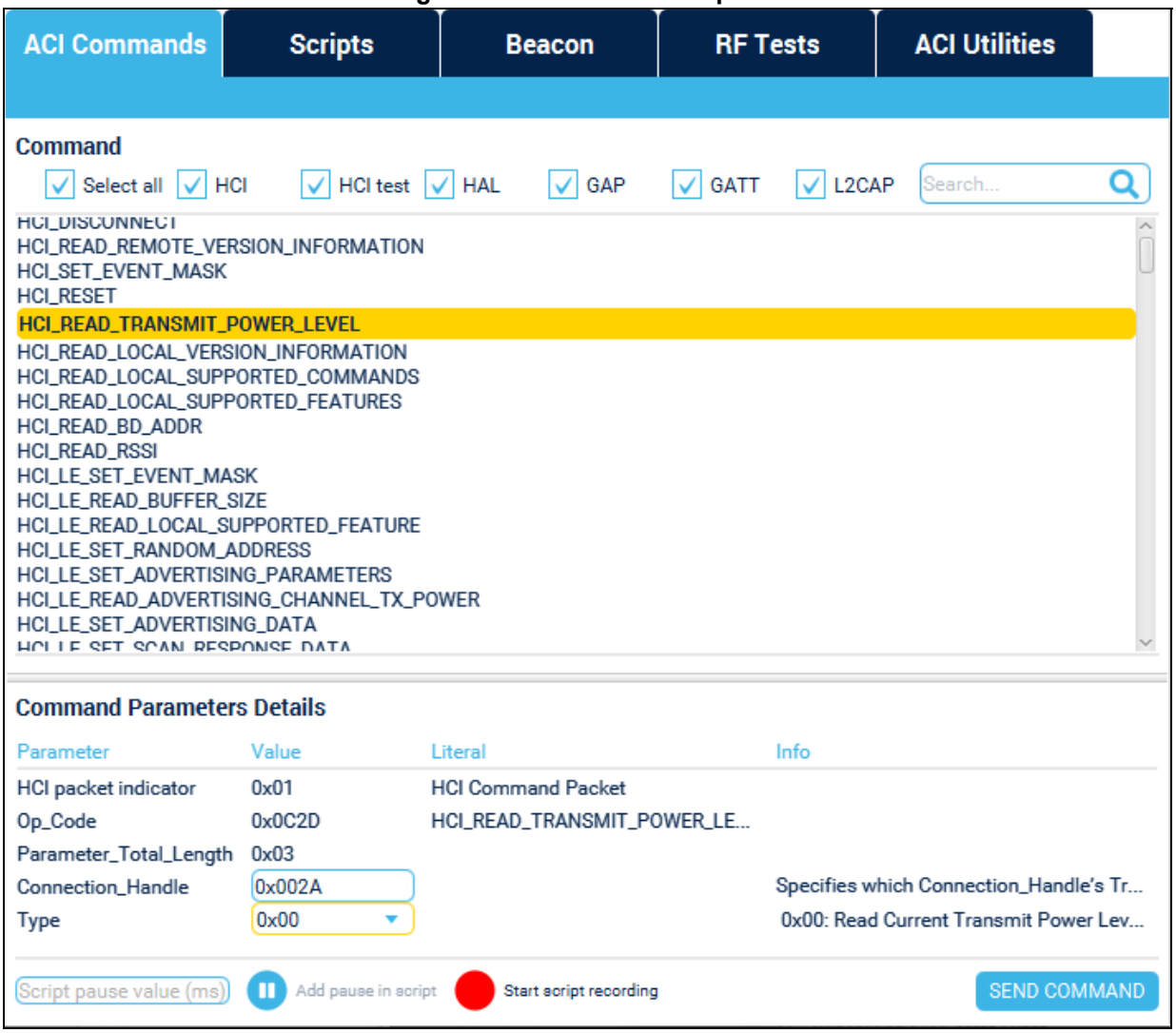
3 Bluetooth low energy mode

3.1 Presentation

3.1.1 Panels

The panels are used to perform a specific operation. Each panel regroups different functions, as *Figure 16* shows it when the *ACI Commands* panel is selected.

Figure 16. ACI Commands panel



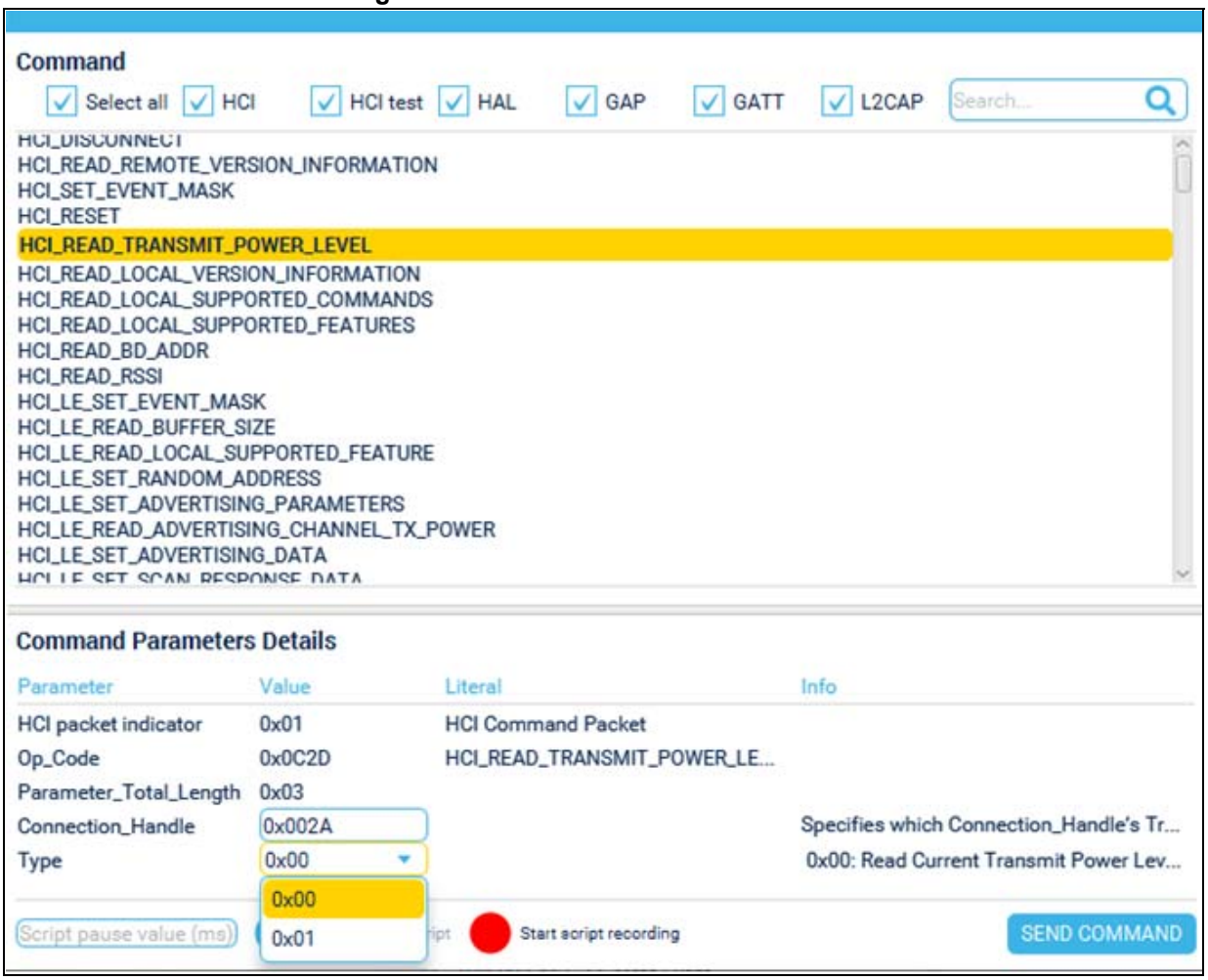
The main panels are *ACI Commands*, *Scripts*, *Beacon*, *RF Tests*, and *ACI Utilities*. They are detailed in the next sections.

3.2 ACI Commands panel

The application command interface (ACI) panel is used to send commands to the main device Bluetooth low energy stack. Commands are grouped by categories. These commands allow the user to configure the Bluetooth low energy stack and activate the communication with remote devices.

3.2.1 How to send an ACI command

Figure 17. How to send an ACI command



Before sending any command to the main device, the device must be connected.

To send an ACI command:

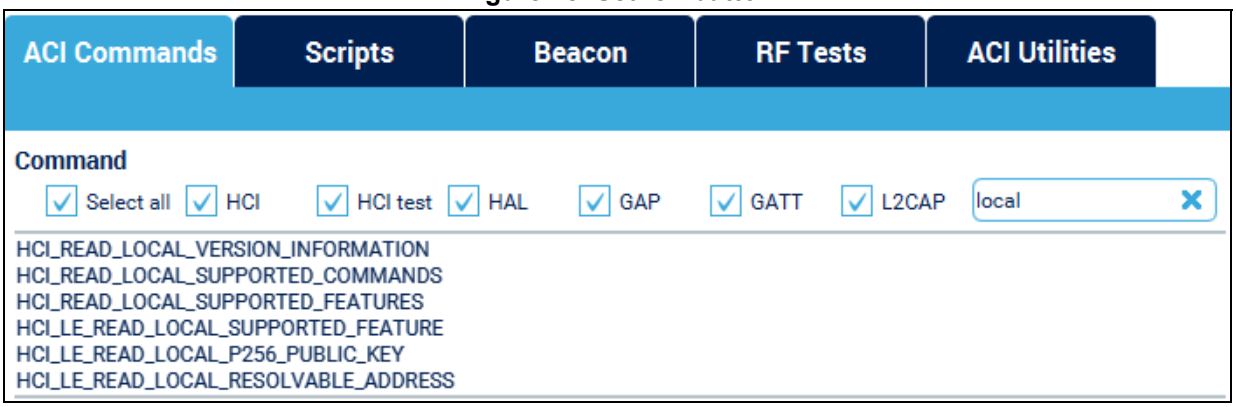
1. Select a command name in the command list (for example HCI_READ_TRANSMIT_POWER_LEVEL).
   The command parameters are displayed in the Command Parameters Details area.
2. Fill in the parameters of the command. Default values are used otherwise.
3. Click on *SEND COMMAND*. The command is sent to the main device

3.2.2 Search function

The search icon is used to quickly select a command in the list:

- Click on the magnifier icon. A text box is created
- Type the name to search. As soon as a character is entered, matching commands are filtered in the list. The match may be any part of the command name, it is no necessary to start from the beginning.
- Click once on the command to select it (Do not use double click).

Figure 18. Search button



3.2.3 Filter usage

The commands are grouped and named by features. Groups are:

- HCI
- HCI test
- HAL
- GATT
- GAP
- L2CAP

The picklist at the top of the area allows seeing only some groups to find more easily the commands. Click on *Select all* to see all commands in the list.

3.2.4 How to fill parameters. Fixed field / editable field

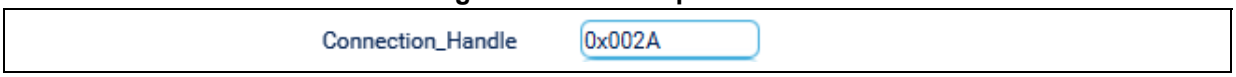
Some parameters have fixed values and are not editable, while others are free or take only some values. The tool guides the user to fill in the parameters:

- Fixed parameter: this parameter is not editable. The value is defined by the specification, or by logic. This applies to the *length* value which is computed by the tool automatically.

Figure 19. Fixed parameter

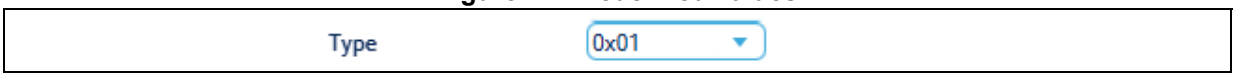
- Editable parameter: the editable parameter is surrounded by a blue rounded box. The value is editable inside the field. Edit is blocked if the value is too long for the field.

Figure 20. Editable parameter



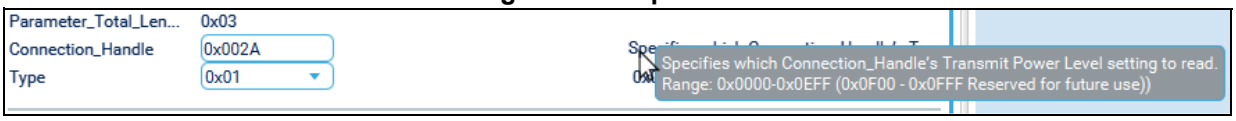
- Predefined values: when the choice is limited, a picklist is displayed to help the user to select the values.

Figure 21. Predefined values



For some parameters, some help is available in the column *Info*. To see the help details, put the pointer on the wanted parameter info, and a bubble displays the details.

Figure 22. Help details



3.2.5 Log functionalities

The log area is on the right part of the screen. It displays the messages exchanged with the boards.

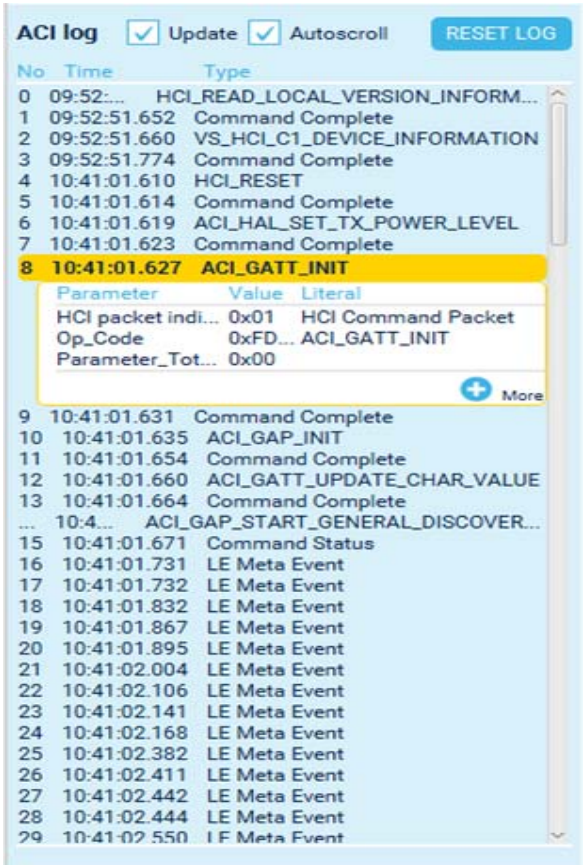
When a command is sent, most of the time an immediate answer comes from the board. It is a command status or a command complete.

The commands with *Command Status* usually have other events coming later. These events are also displayed in the log area.

Some asynchronous events may come from the device and be displayed in this area.

The tool keeps the last 1000 lines. When the limit is reached, the oldest lines are automatically discarded.

Figure 23. Log functionalities



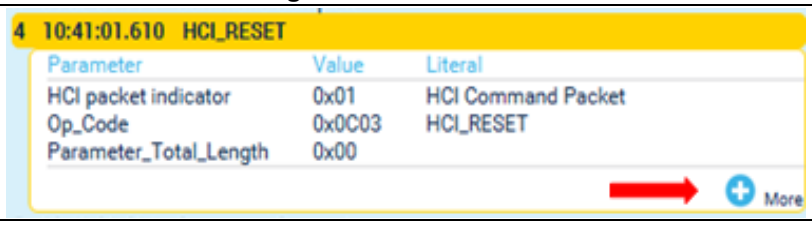
It is possible to scroll in the list with the scroll bar on the right side.

When a line is selected, the content of the selected message is displayed in the green area, with one line for each parameter.

The text ends with … when it is not possible to display complete text. It is possible to change the log area width to display longer texts.

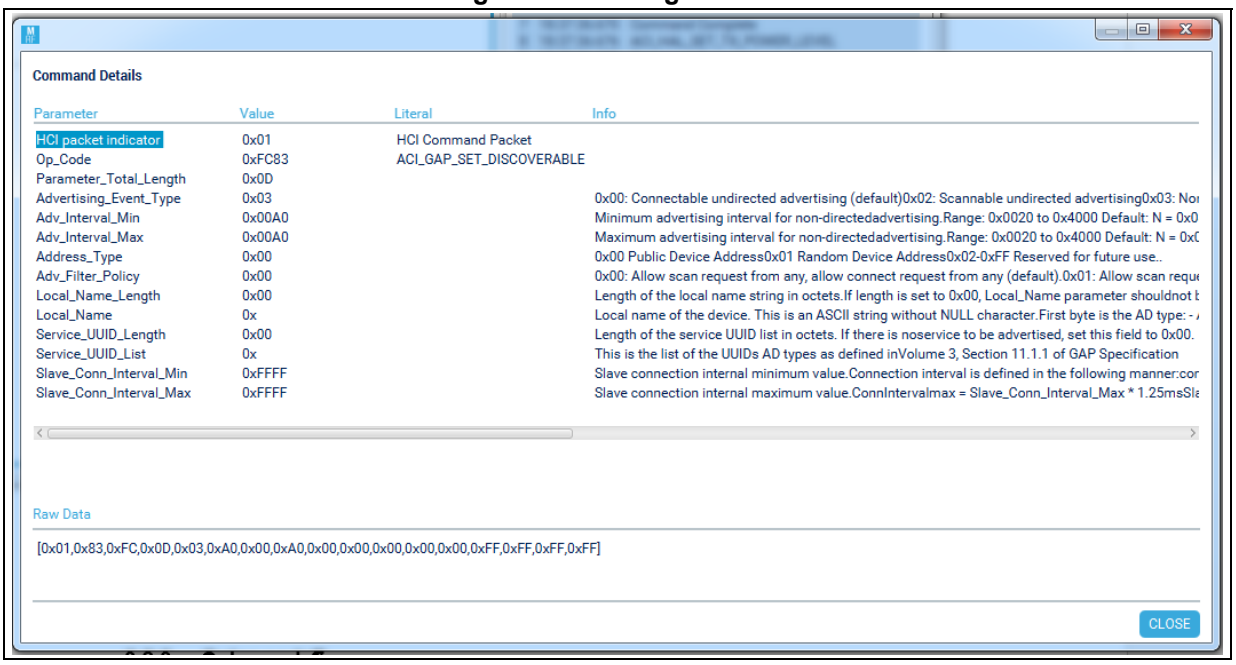
Details

Figure 24. More button



Sometimes, all the information of a message does not fit in the area used for the log. The button *Figure 24* opens a new window showing the message details:

Figure 25. Message details



The details show all decoded message parameters. The *Literal* column shows predefined text for the parameter values (Opcode and others). The *Info* column provides some description of the parameter content.

The raw data in the bottom part is the data sent/received over UART, without decoding.

In this window, it is possible to copy information for pasting it into other windows.

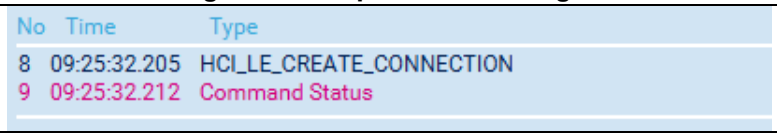
An efficient solution to compare two messages is to open multiple detail windows at the same time.

Color code

The logs use color code to identify the device used and highlight errors.

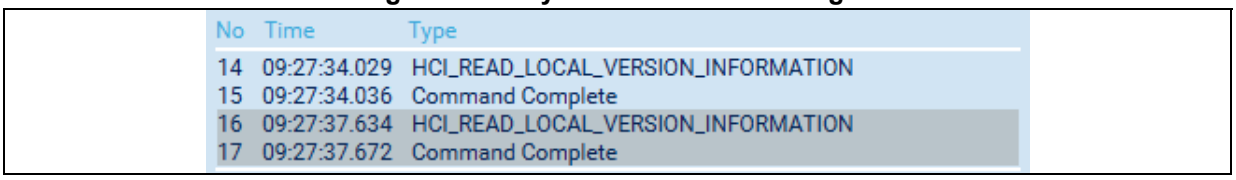
A line with purple text shows that the status in the message is different from zero, which indicates an error.

Figure 26. Purple error messages



Log on a dark gray background is coming from a second board. When two boards are connected, the main device (DUT) has a normal color log while the second device tester has a darker background. This is helpful to understand the sequences involving two devices.

Figure 27. Gray second board messages



Update button

When the *Update* tick box is not selected, the messages are not added in the log area. The line number continues to be increased anyway but is not displayed until the *Update* tick box is enabled.

Auto-scroll

When the *Auto-scroll* box is ticked, the log area always displays the last log received. To check the log history, untick the box which disables the auto-scroll.

Reset Log

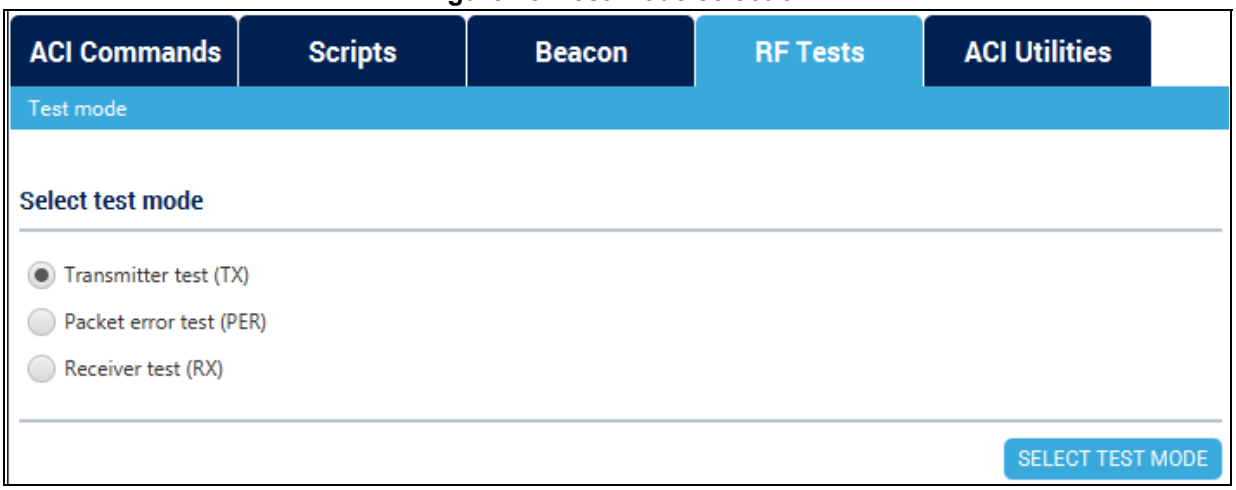
The *RESET LOG* button allows wiping the log displayed in the log area. The line number is not affected, but the memory used by older logs is made free.

3.3 RF test panel

The RF test panel is used to perform the radio-frequency tests on the main device. The RF tests are grouped into three test modes: Transmitter (TX), Receiver (RX), and Packet error rate (PER):

- The TX test is dedicated to radio-frequency emission, for tones and packets.
- The RX test is for packet reception.
- The PER test is a quality-transmission test between two devices.

Figure 28. Test mode selection



The first action after connecting a device is to select the mode to test and then to click on the *SELECT TEST MODE* button.

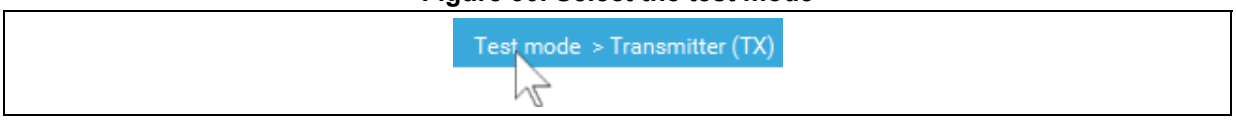
When the user has selected a test mode, it is mandatory to go back to the selection page to change the test mode:

- Click on the *Change test mode*

Figure 29. Change the test mode

- Click on *test mode* in the top bar.

Figure 30. Select the test mode

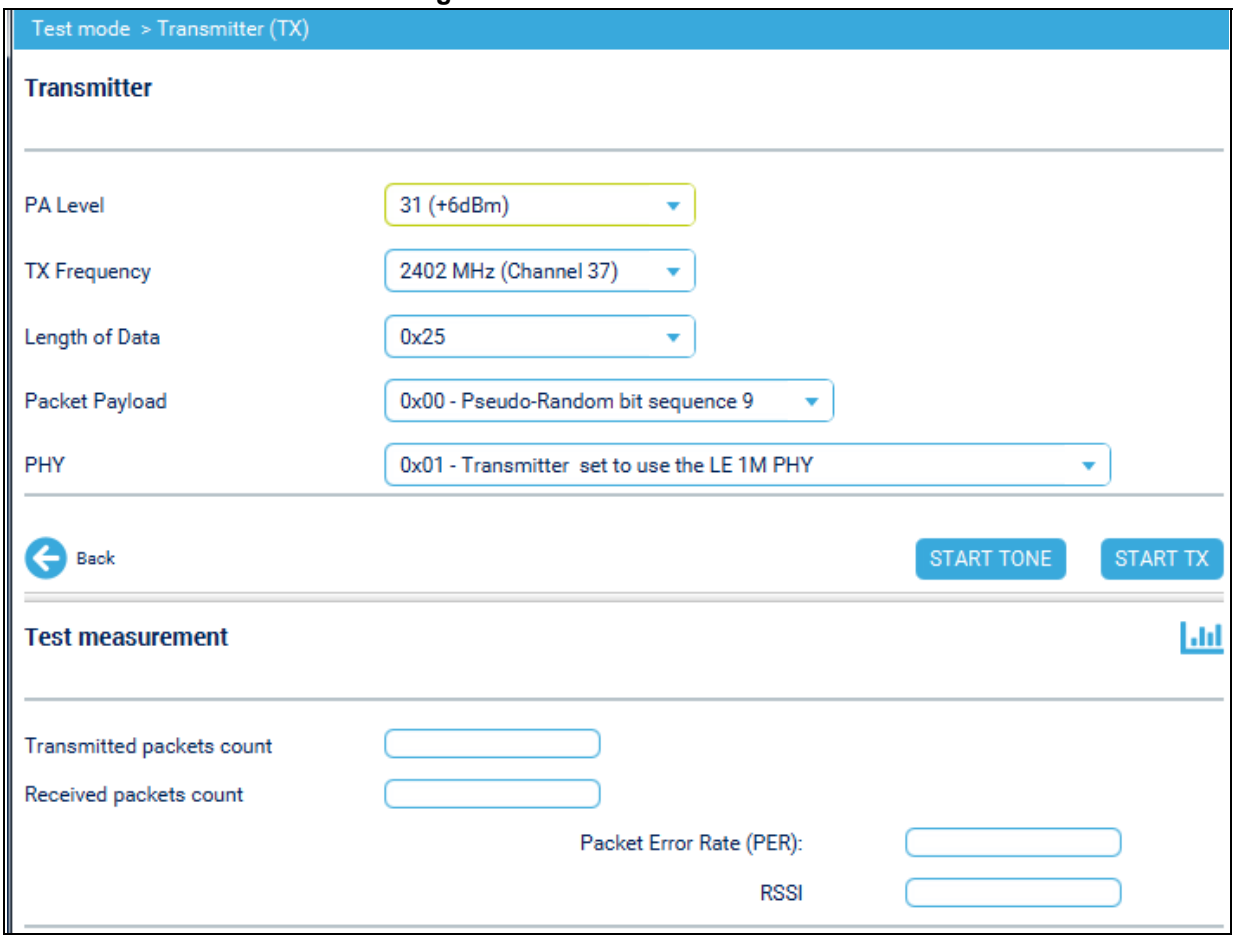


Note: To avoid incorrect configuration of the device, the test mode is unchangeable, when transmission or reception is ongoing. The user must first stop the transmission and then change the test mode.

3.3.1 Test mode transmitter (TX)

The TX mode is used to set the Bluetooth low energy transmitter in emission. Two transmission modes are defined: *transmission of data*, or *emission of tone*.

Figure 31. Test mode transmitter



Tone generation

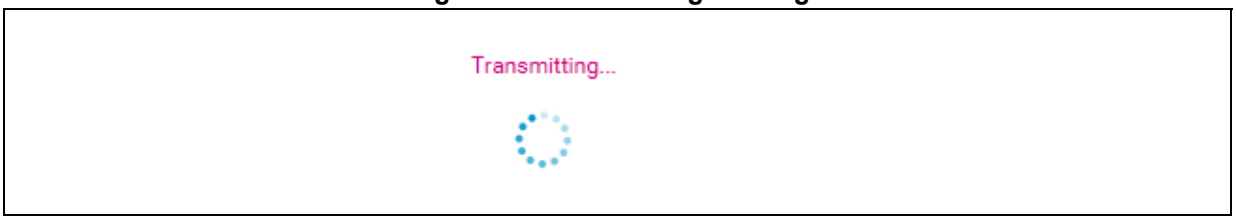
The tone generation performs the emission of a continuous sinus wave on the RF. The parameters for the tone are tone power level and tone frequency. The power level is the power at the chip output.

To start tone generation:

1. Enter the Transmitter panel test Mode
2. Select the power level with the picklist.
3. Select the frequency with the TX Frequency picklist. The list is sorted by frequency; the data/advertising channel index is indicated in parenthesis. The advertising channel index does not follow the frequency order. Channels 37, 38, and 39 are the advertising channels. Refer to BLUETOOTH SPECIFICATION Version 4.2 [Vol 6, Part B] ch1.4.1 for details.
4. Select the PHY modulation to use (The modulations not supported by the device are not listed).
5. Click on the *START TONE* button.

The emission starts, the *START TONE* button is changed to *STOP TONE*, and *Transmitting information* is displayed:

Figure 32. Transmitting message



6. To stop the tone generation, click on *STOP TONE,* and the emission stops.

It is mandatory to stop transmission to change to another test mode.

Packet transmission

It is possible to send some data packets in test mode. The parameters are power level, transmission frequency, length, and content of the data to send.

Power and level parameters are the same as tone parameters.

The packet data is selected in the Packet payload picklist. Eight types of payloads are available:

- A pseudo-random bit sequence 9 (PRBS9)
- A pattern of alternating bits *0b11110000*
- A pattern of alternating bits *0b10101010*
- A pseudo-random bit sequence 15 (PRBS15)
- A pattern of fixed bits *0b11111111*
- A pattern of fixed bits *0b00000000*
- A pattern of alternating bits *0b00001111*
- A pattern of alternating bits *0b01010101*

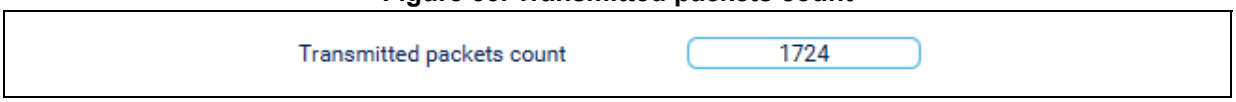
The sequence length is defined by the *Length of data* picklist. This is the length of the data payload in bytes. The PHY box is used to select the modulation.

To start packet emission:

- Select the power level with the picklist.
- Select the frequency with the TX Frequency picklist.
- Select the length of the packet to send
- Select the content of the packet payload
- Click on *START TX*
  The emission starts, the start button is changed to *STOP TX*, and Transmitting is displayed. The sequence is repeated until the test is stopped

To stop the transmission, click on *STOP TX*. The number of packets transmitted during the test is displayed in the test measurement area.

Figure 33. Transmitted packets count

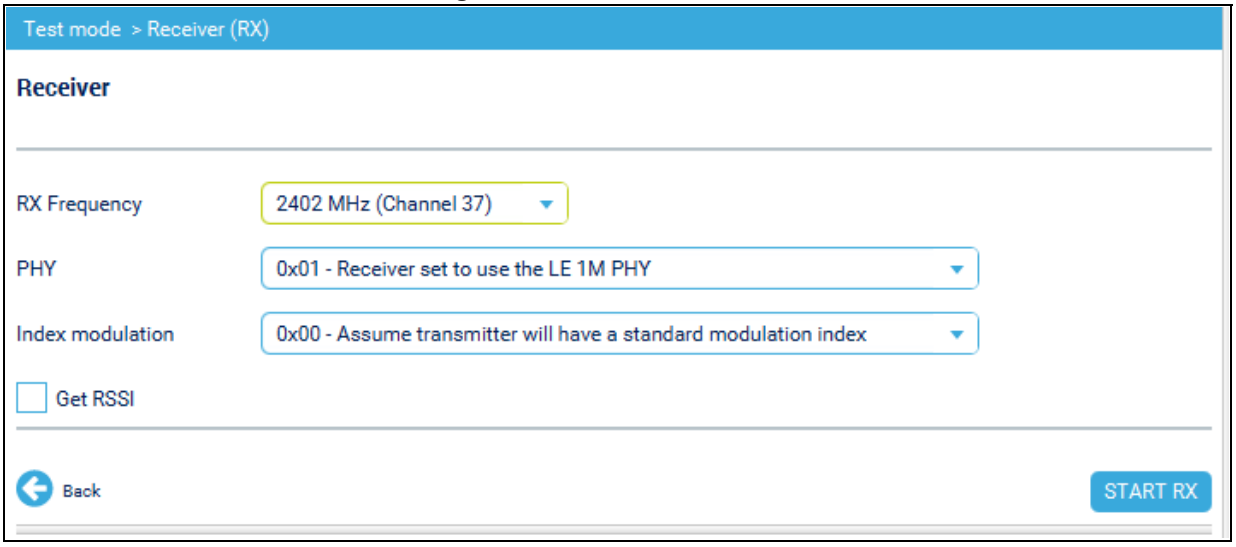


If the number of packets received by the reception device is known, it is manually entered in the *Received Packet Number* box, and the Packet Error Rate is automatically computed (Refer to *Section 3.3.3: PER* for details).

3.3.2 Test mode receiver (RX)

The receiver mode is used to put the main device in reception mode and count packets received.

Figure 34. Test mode receiver



Packets reception:

- Select the frequency to use.
- Select the PHY and the modulation index to be used.
- Click on *START RX*. The reception starts, *Receiving* is displayed with an animation and button change to *STOP RX*.

To stop reception, click on *STOP RX*. The count of received packets is retrieved from the main device and displayed in the *Received packet number*.

If the number of transmitted packets is known, it may be entered manually in the *Transmitted packet number*. The *Packet error rate (PER)* is automatically computed (Refer to *Section 3.3.3: PER* for details).

If the *Get RSSI* checkbox is selected, the tool performs RSSI measurement.

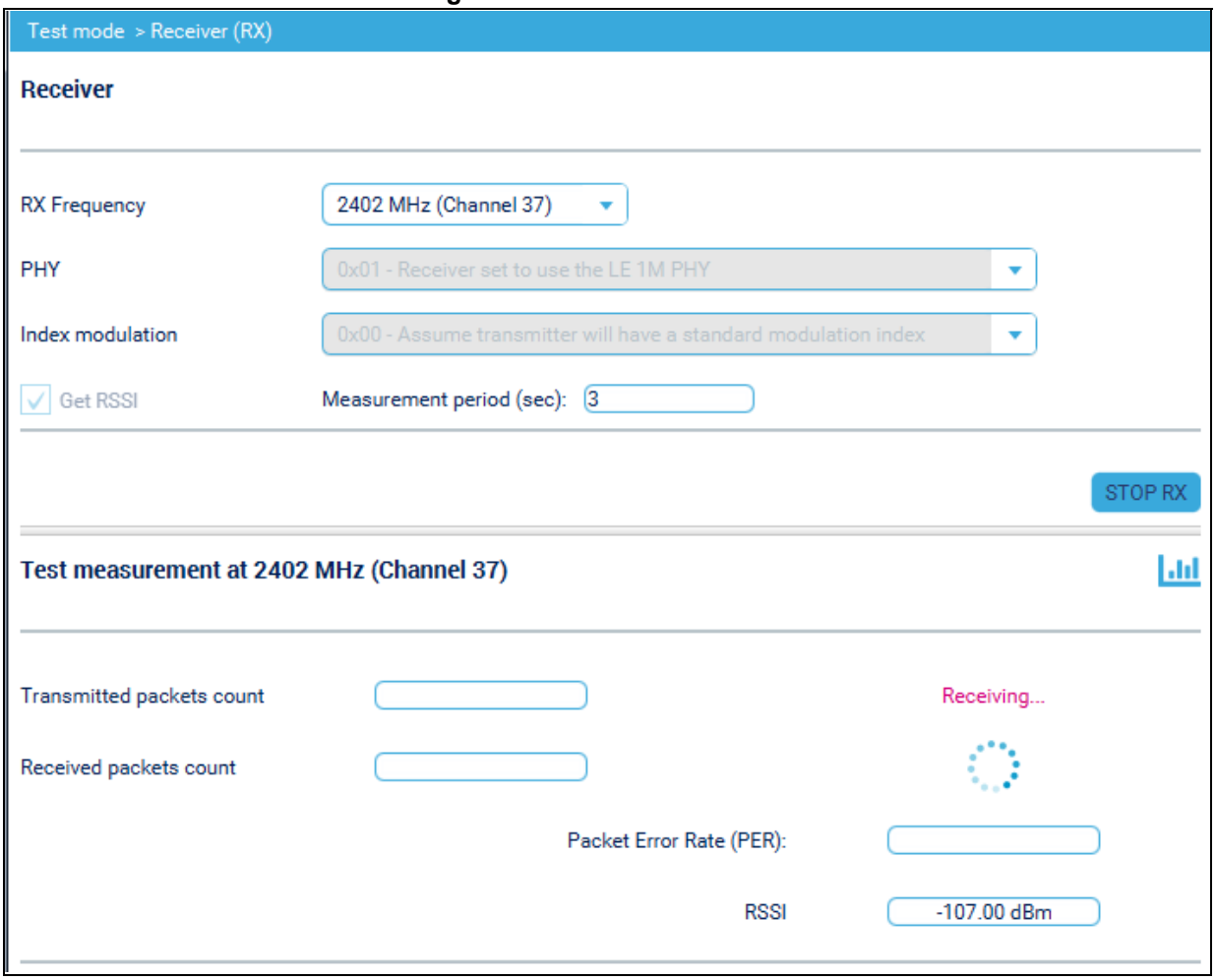
RSSI measurement:

The RSSI indicates the signal level received by the RF. The value reported by the RF is not an absolute value because the reception level is dependent on the board layout and antenna design.

When the RSSI option is selected, the user must define the measurement interval. The default value is 3 seconds. The RSSI value is displayed at the end of each measurement period.

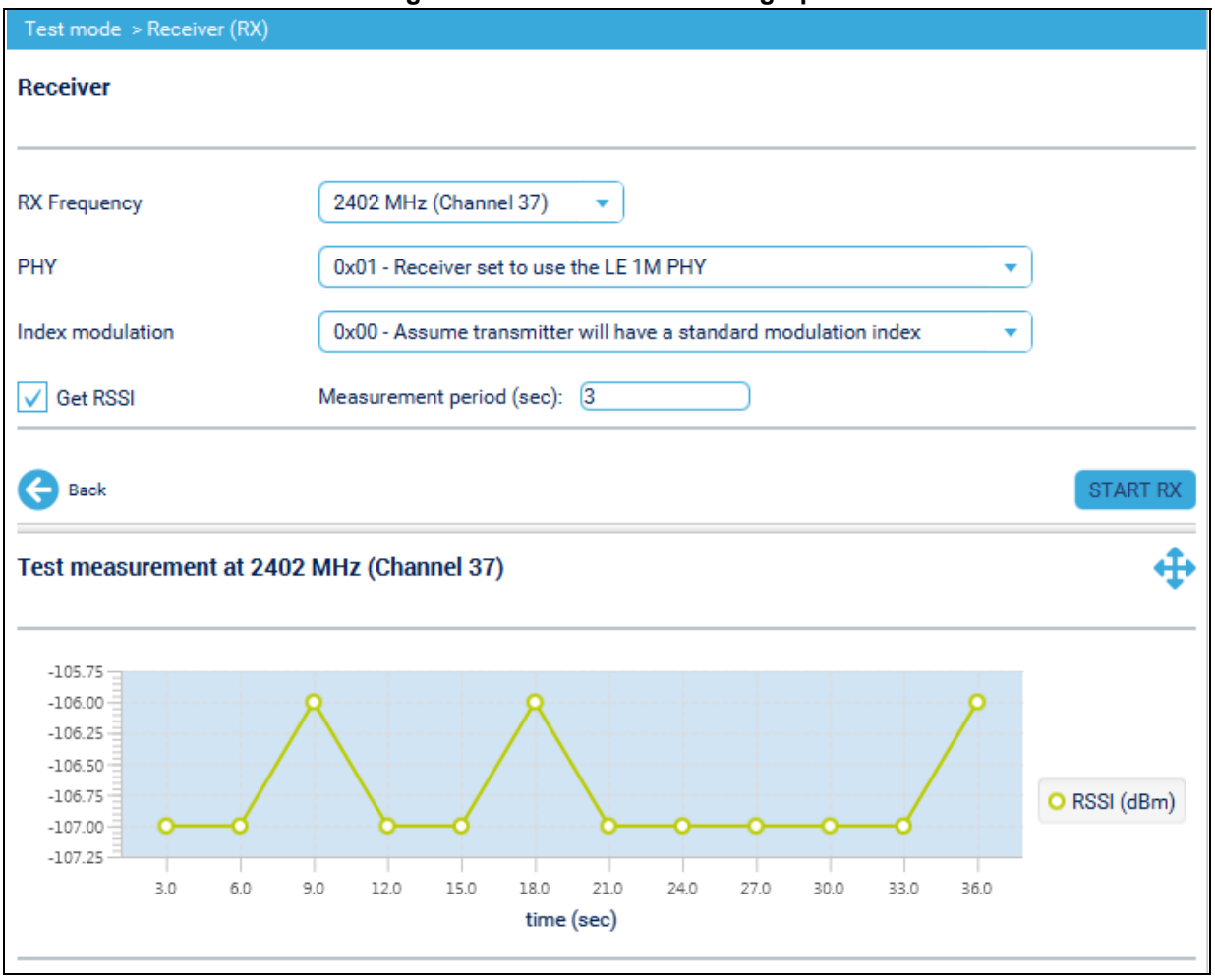
It is possible to switch between detailed value, plot view, and big display, with the blue button on the right (bar chart, arrows, or blue lines).

Figure 35. RSSI measurement



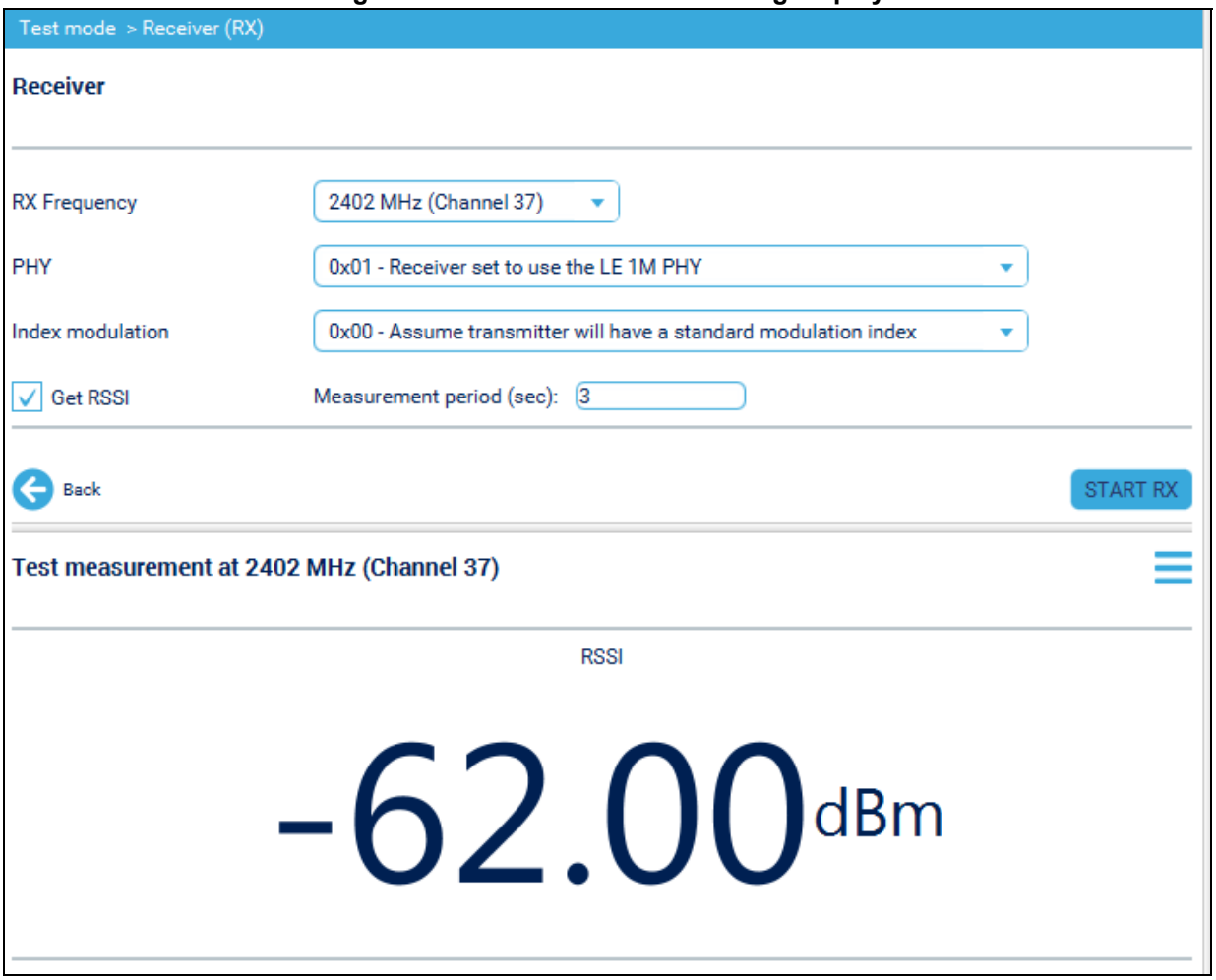
Note: When the RSSI measurement is performed, the number of received packets is not available in the tool. When the measurement is stopped, the Received packet number field is cleared, and an information message is displayed.

Figure 36. RSSI measurement graph



Note: The graph length is limited to 250 points. When the limit is reached, the oldest points are discarded.

Figure 37. RF RSSI measurement big display



3.3.3 PER

PER definition

The packet error rate (PER) is an indicator of the quality of transmission between two devices. The measurement proposed in the tool covers the whole transmission chain from the transmitter to the receiver.

The packet error rate is computed with the number of packets sent and the number of packets received. A good transmission gives a low PER. High PER means that transmission is not good.

Figure 38. PER definition

$$PER = 100 \times \frac{Ntx - Nrx}{Ntx} \%$$

Ntx: number of packets sent, Nrx number of packets received, PER result in percent.

A bad PER may be an issue from the transmitter or the receiver and depends on parameters like the distance between devices, antennas, PCB design, and interferences. To limit the parameters influencing the measurements, it is advised to use one reference board with well-known performances in the setup.

PER test mode

The tool provides a special test mode dedicated to the PER test. In this mode, two devices need to be connected to the computer:

- the first device under test (DUT)
- the second device used to act as a packet generator (tester)

After the connection of the DUT (main device, connected in the application top bar), the PER test mode is selectable on the RF test page.

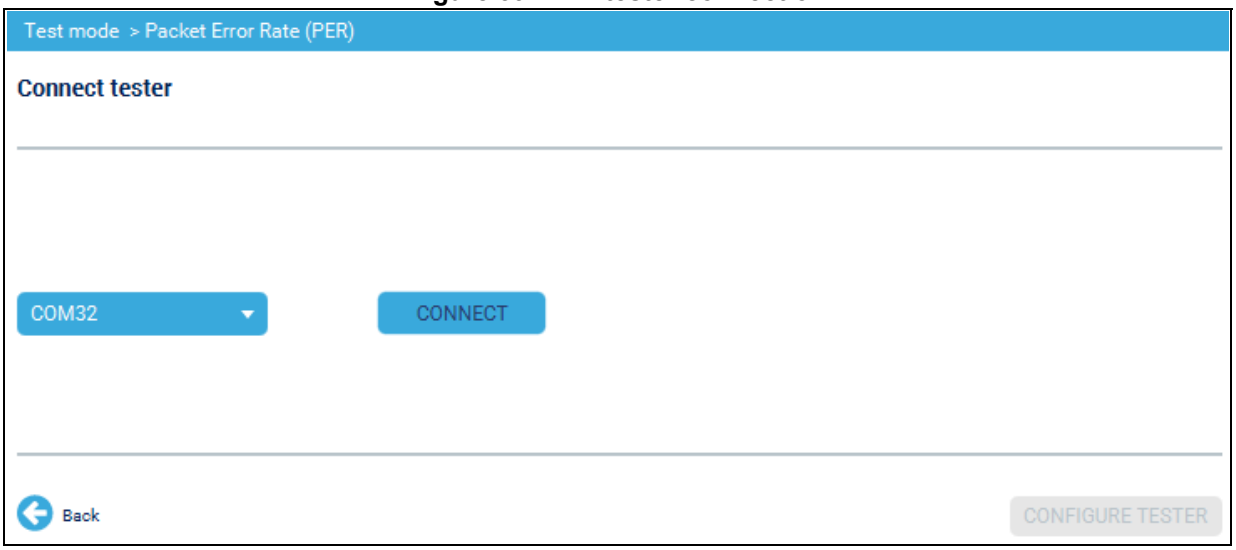
The configuration of PER test is done with a sequence of panels:

- tester connection
- tester configuration
- DUT configuration
- test parameters

The first step is to connect the tester:

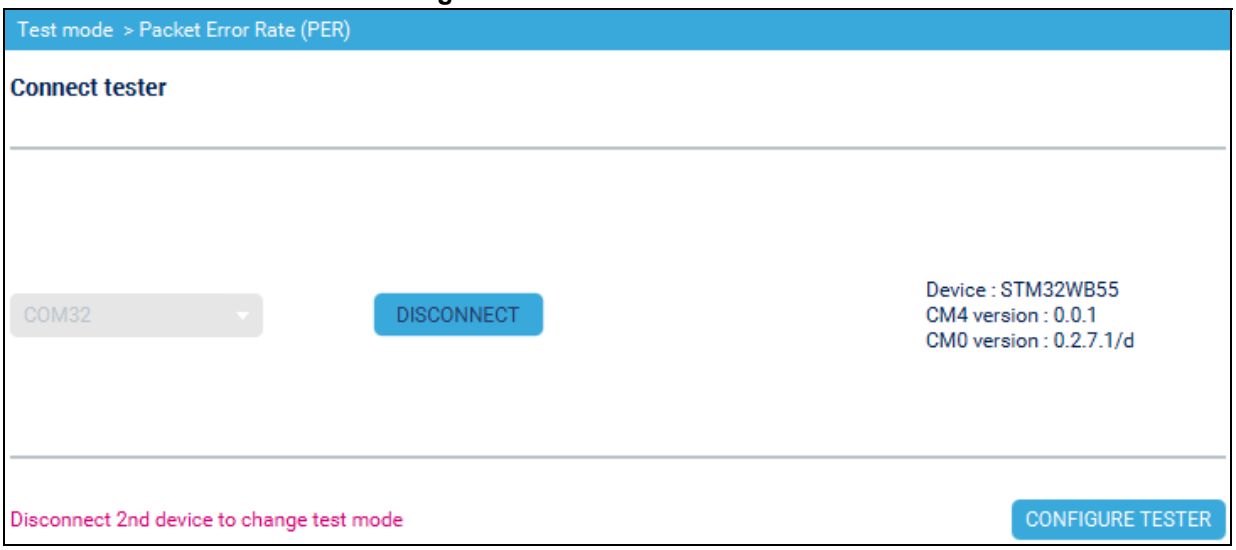
PER tester connection

Figure 39. PER tester connection



- Plug the device in the computer (same requirements as the first device, refer to *Chapter 2.2*).
- Select the serial port to use in the picklist.
- Click on the *CONNECT* button.

Figure 40. PER tester connected

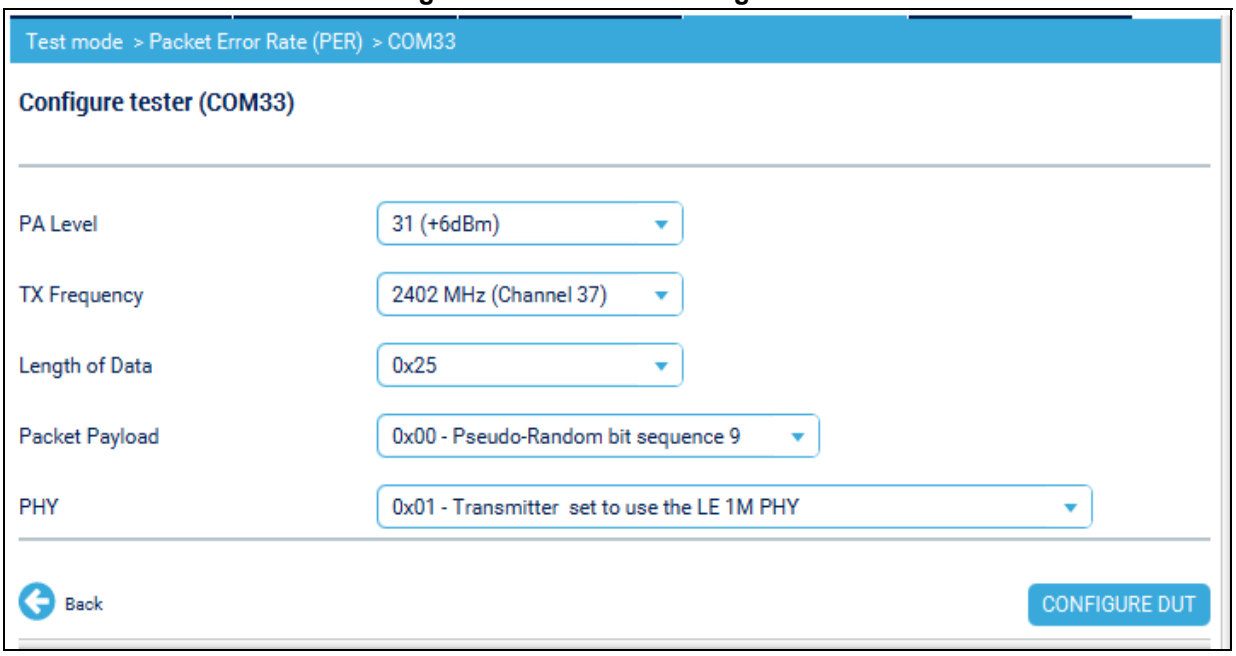


- The board information is displayed on the right.

When the second device is connected, it is not possible to change mode. Disconnect the device first, and then use the *back* button.

Click on *CONFIGURE TESTER* to set the tester parameters:

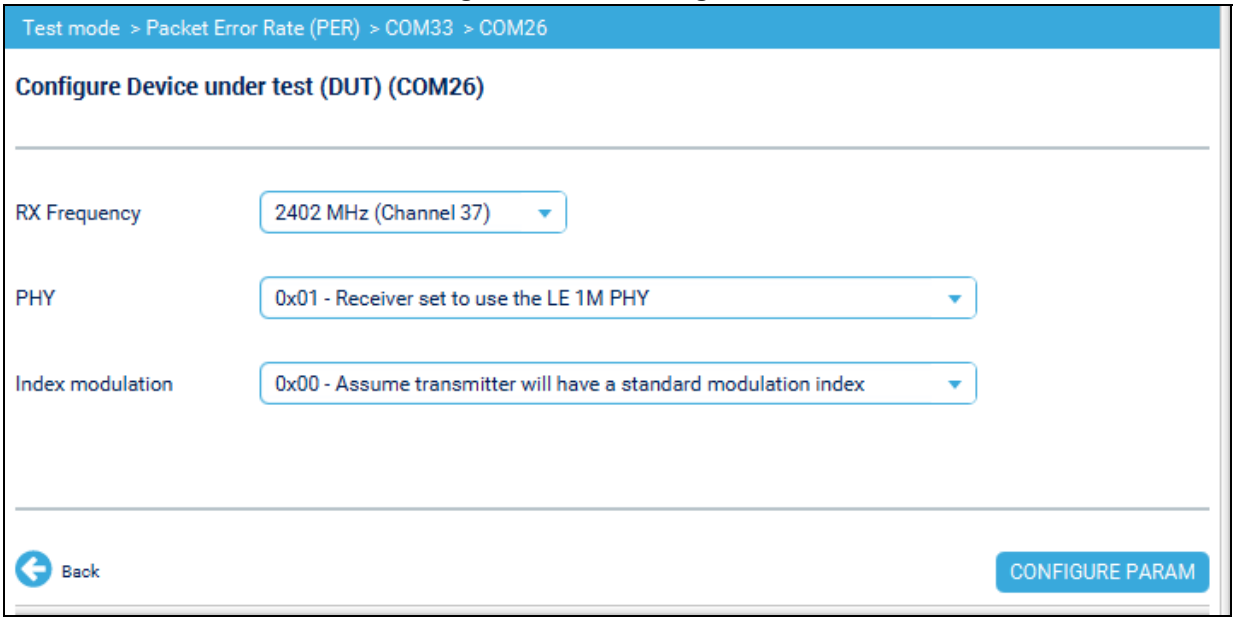
Figure 41. PER tester configuration



- Select the TX power level with the picklist.
- Select the transmission frequency with the *TX Frequency* picklist.
- Select the length of the packet to send (same as the TX test).
- Select the content of the packet payload. For PER test, it is recommended to use the *Pseudo-Random bit sequence 9* reference pattern. Patterns with only 0 or 1 bits must not be used for PER. The other patterns can be used.
- Select the PHY to use.

Click on *CONFIGURE DUT* to set the Device Under Test configuration:

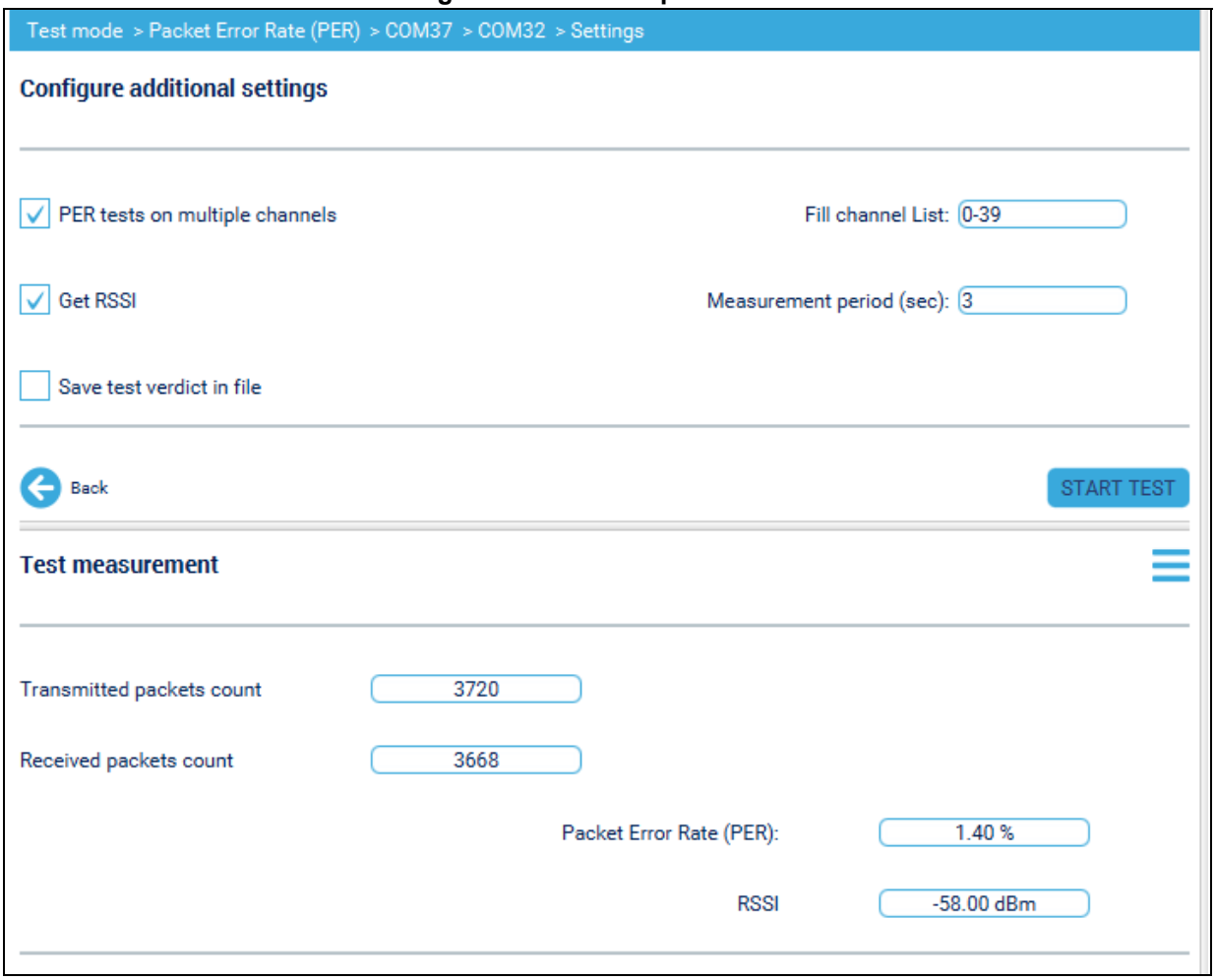
Figure 42. DUT configuration



Select the reception frequency, the PHY, and the modulation index for the receiver board. The tool uses by default the same frequency as the tester, but the user may modify it.

Click on *CONFIGURE PARAM* to set the test configuration:

Figure 43. PER test parameters



- *PER tests on multiple channels*: when this option is selected, the PER test is performed on a list of predefined channels. When the box is ticked, the Channel list is displayed. Value 0-39 indicates all channels between 0 and 39. It is also possible to put value separated by a comma: *0,1,5* or to mix: *0,1,10-15*.
  The measurement period is the time of each PER test to be performed.
- *Get RSSI*: this option adds some RSSI measurement between each PER measurement. When it is activated, the tool performs a PER test for the Measurement period, computes PER, and then makes an RSSI check.
- *Save test verdict in file*: this option generates a test report of the measurements. When the option is selected, a *SELECT FILE* button is displayed. The user must select the report file before starting the tests. The report is saved at the end of the tests.

When the option has been configured, click on the *START TEST* button:

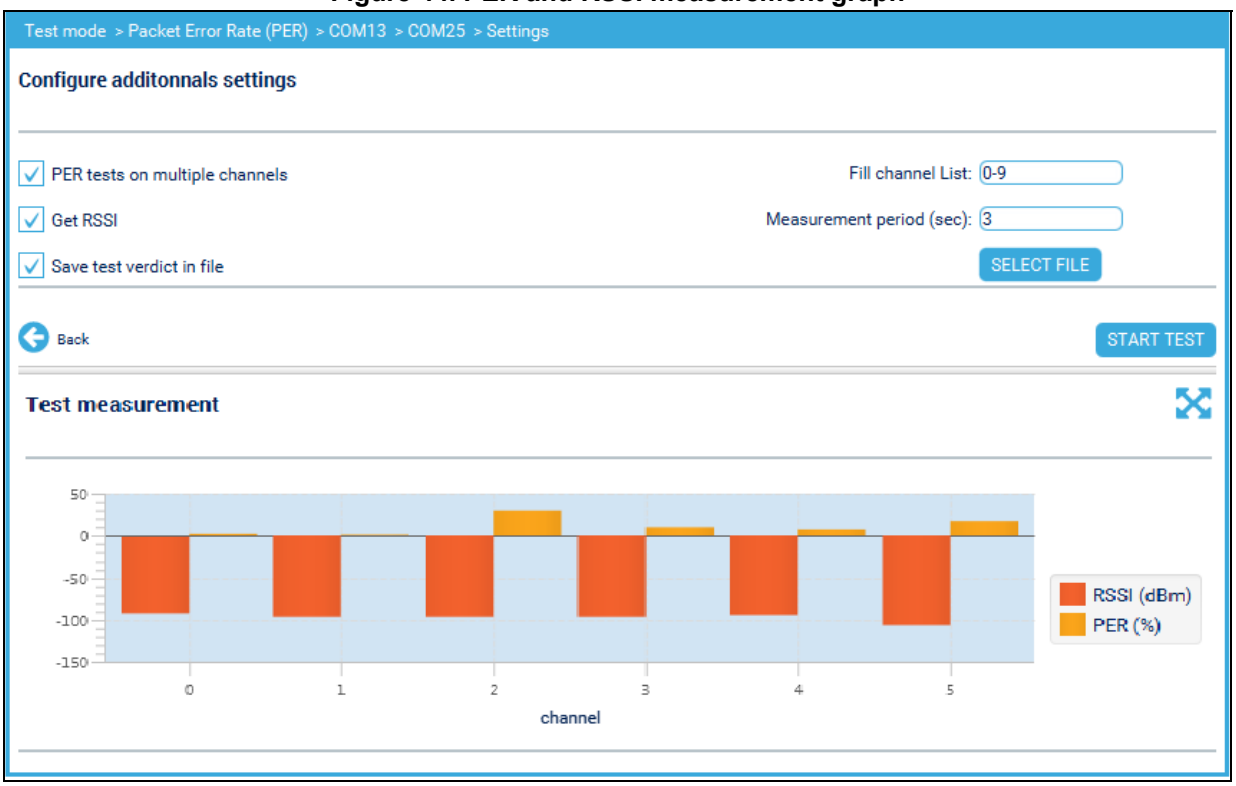
- the DUT is set in reception mode,
- then the tester starts.

The button is changed to *STOP* and *Testing…* is displayed.

The test continues until the user presses the *STOP* button, or when all channels have been measured for multichannel tests.

The result is displayed in the bottom part. It is possible to switch between the numerical values and the chart with the blue bar icon.

Figure 44. PER and RSSI measurement graph

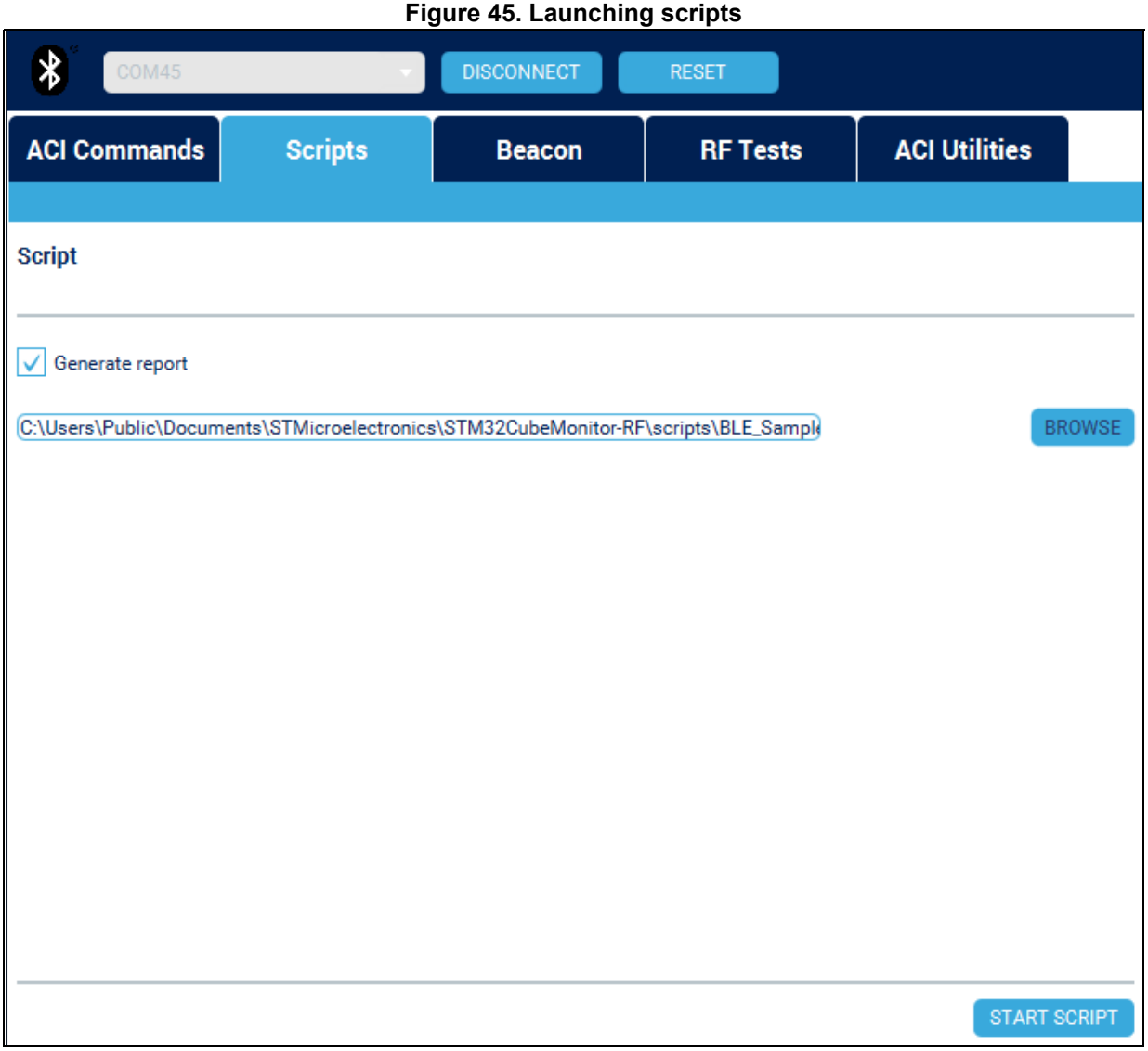


3.4 Scripts

Scripts are used to execute in sequence some commands stored in a text file. Scripts help avoid entering each command manually for repetitive tasks.

3.4.1 Launching scripts

Figure 45. Launching scripts

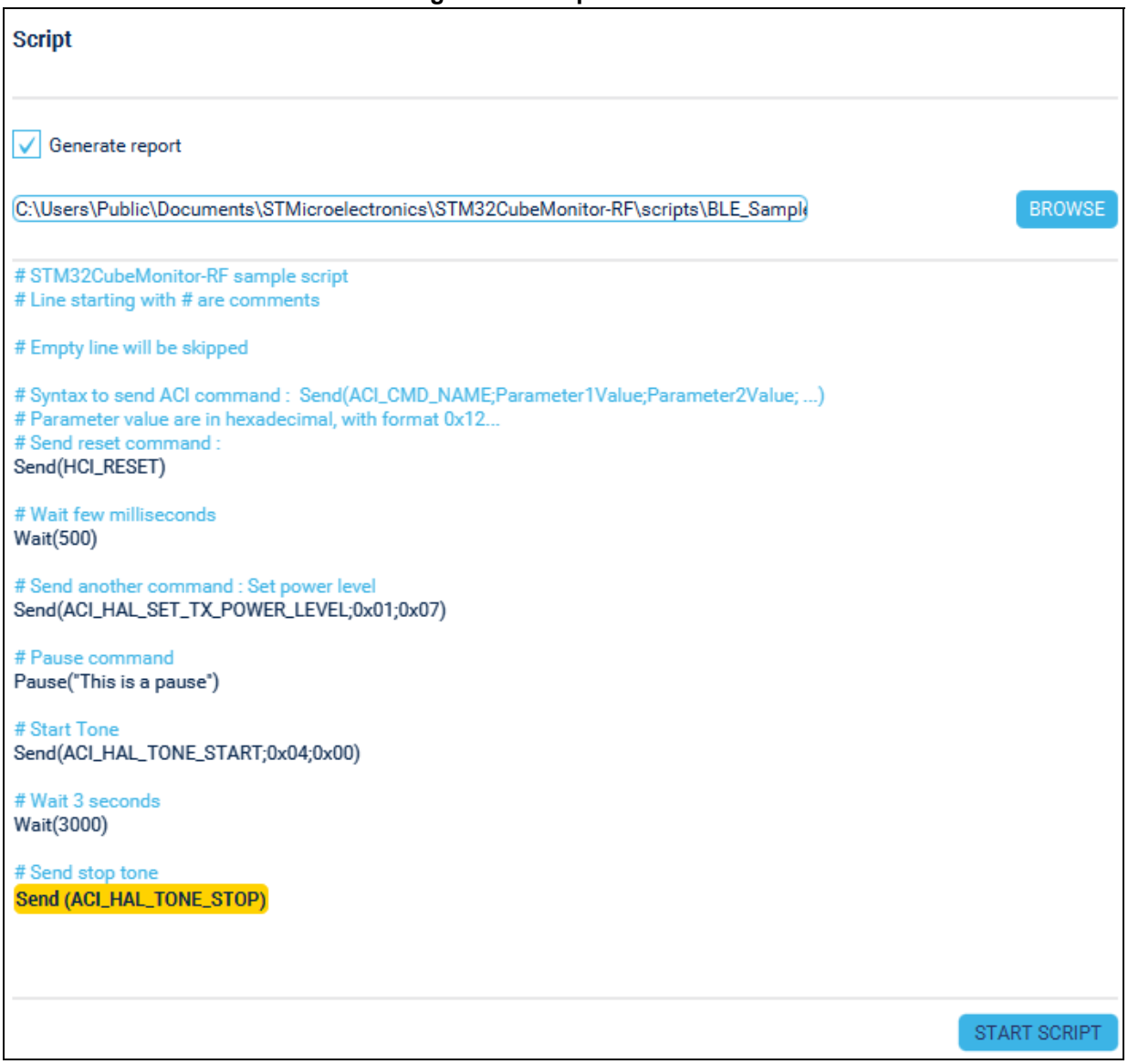


Scripts are stored in text files and are editable with any text editor.

To execute a script:

- Select the script file with the browse button or directly enter the file name.
- Click on the *Start script* button.
- The script is displayed and executed. The line in execution is highlighted in green. The ACI results are updated in the log area.
- The script is manually stopped with the *Stop script* button.

Figure 46. Script execution



Script examples are provided with the tool, such as sample script, loop, beacon creation.

For Windows, scripts are in the folder *<Documents>\STMicroelectronics\STM32CubeMonitor-RF\scripts* (public documents).

For Linux, they are in *<userhome>/STMicroelectronics/STM32CubeMonitor-RF/scripts*.

For macOS, it is inside the document folder provided in the setup package.

3.4.2 Script recording

The ACI commands used in the ACI panel are directly recorded in a script. Some script buttons are located at the bottom of the ACI panel:

Figure 47. Script recording buttons

Use the red button to start recording. Pause is inserted with the *Add pause in script* button.

At the end of the recording, click the *Stop* button. The tool asks the script name before saving.

3.4.3 Scripts modification

The script is created or modified with a text editor. It uses a simple syntax to list the ACI command to send and the action to perform.

Figure 48. Sample script

```
# Send reset command :
Send(HCI_RESET)

# Wait few milliseconds
Wait(500)

# Send another command: Set power level
Send(ACI_HAL_SET_TX_POWER_LEVEL;0x01;0x07)

# Start Tone
Send(ACI_HAL_TONE_START;0x00)

# Wait 3 seconds
Wait(3000)

# Send stop tone
Send (ACI_HAL_TONE_STOP)

# Pause command
Pause("End of script")
```

The lines starting with # are comments and are ignored by the tool. Empty lines are skipped.

Other lines are commands. The line starts with the command name, followed by parameters in brackets separated by a semicolon.

3.4.4 Script report

It is possible to have a script report generated at the end of script execution. The script report stores the status of each ACI command executed by the script.

Figure 49. Script report

```
SCRIPT REPORT

Script name : SampleScript.txt

Test date : 18/12/2017 17:27:52

Verdict : SUCCESS    No error detected


Command                       | Sent | ACI status | ACI raw result

HCI_RESET                     | OK   | 0x00       |
[0x04,0x0E,0x04,0x01,0x03,0x0C,0x00]

ACI_HAL_SET_TX_POWER_LEVEL    | OK   | 0x00       |
[0x04,0x0E,0x04,0x01,0x0F,0xFC,0x00]

ACI_HAL_TONE_START            | OK   | 0x00       |
[0x04,0x0E,0x04,0x01,0x15,0xFC,0x00]

ACI_HAL_TONE_STOP             | OK   | 0x00       |
[0x04,0x0E,0x04,0x01,0x16,0xFC,0x00]

END of report
```

The result is stored in a new file, in the same path as the script, with a name in the form: *verdict_SampleScript_18-12-2017_17-27-52*. The name is built with the concatenation of:

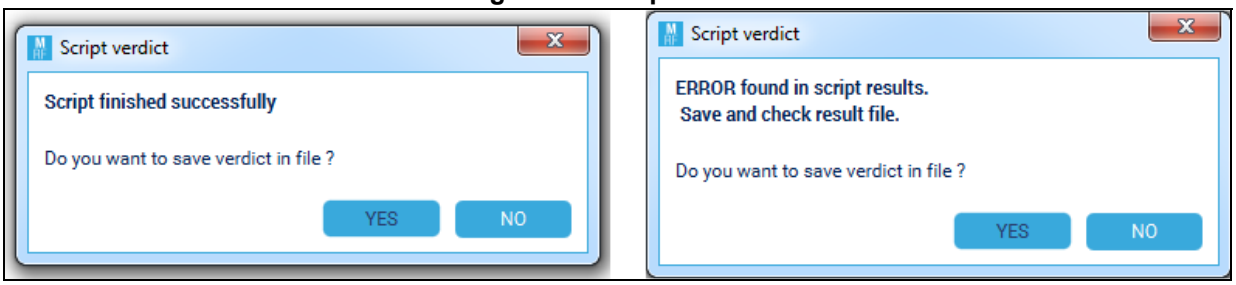
- *verdict_*
- script name
- current date
- current time
- .txt extension

In the report, the *Sent* column holds the status of command transfer to the board. If parameters are missing, the command is not sent.

The *ACI status* column has the status of the ACI response. *0x00* is a success status, other values are errors.

At the end of script execution, a popup with the verdict (error found or finish successfully) is displayed and asks if the report must be saved:

Figure 50. Script verdict



If the user presses *yes*, the report is generated in the folder of the current script. If the user presses *no*, the report is not saved.

If the *Generate report* tick box is not checked, no report is generated at the end of the script. The script successfully means there was no error in the script syntax, and the status of operations was *OK* (error code = 0). The value measured and the performance are not verified, there is no PASS/FAILL criterion on the results.

3.4.5 List of script commands

Send an ACI command:

The ACI commands are sent with the instruction *Send*: `Send (ACI_CMD_NAME; Parameter1Value; Parameter2Value...)`

The elements inside the parenthesis are separated by semicolons.

The first element is the command name. It is the name as it is displayed in the tool.

The next elements are the parameters. The value must be entered in hexadecimal format and start by *0x*. The optional parameters can be left empty. The length is dependent on parameter size in the ACI command.

Note: The Command Packet Type, Opcode, and Parameter Total Length are filled by the application. They must not be added to the parameters.

Wait for a specific time

It is possible to add a delay with the instruction *Wait*:

```
Wait (3000)
```

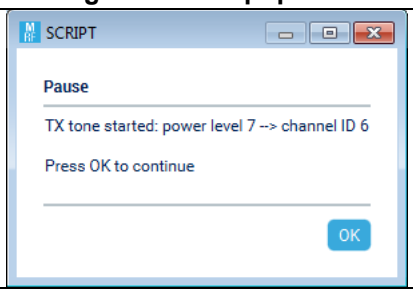
This instruction delays the script execution for 3 seconds. Time is given in milliseconds.

In the ACI panel screen, a pause is inserted in the script with the *Add pause in script* button.

Pause command in the script

The Pause command adds a pause during the proceeding of the script. This command opens a pop-up window customized with the user comment.

Figure 51. Script pause



The *OK* button allows continuing the script.

Command: Pause (*User comment*)

The user text must be enclosed between quote marks (“).

Figure 52. Example

```
# Pause demo script


# Start Tone
Send(ACI_HAL_TONE_START;0x04)


# Pause command
Pause("TX tone started")


# Send stop tone
Send (ACI_HAL_TONE_STOP)
```

Loop command in the script

Loop can be used in the script to repeat some actions automatically.

Loop usage

To repeat a part of a script, the commands must be enclosed between 2 instructions:

- Loop (count,0,5); this instruction indicates the beginning of the repeated section. *count* is the name given to the counter, the first value is the start value, and the second one is the end value. In this example, the counter count is being increased from 0 to 5; there are 6 iterations.
- EndLoop: indicate the end of the loop. If the counter reaches the end value, execution continues the next line. If the counter has not reached the end value, the counter is updated, and execution goes back to the *Loop* instruction.

Figure 53. Loop simple example

```
Loop (count; 1; 3)
Pause ("test the loop")
EndLoop
```

This script, given as an example in *Figure 53*, displays *test the loop* three times.

Using the counter value

It is possible to use the counter value in other lines of the script to change the parameter values during script execution. When the counter name is embedded inside square brackets, the tool inserts the counter value.

Figure 54. Loop second simple example

```
Loop (count ; 1 ;3)
Pause ("The loop counter is [count]")
EndLoop
```

The script in *Figure 54* displays *The loop counter is 1*, then *The loop counter is 2*, and finally, *The loop counter is 3*.

Some parameters require hexadecimal values. In this case, add an ampersand (&) after the first bracket. The tool replaces the counter name with the hexadecimal value.

If count = 10, [&count] is replaced by *0xA*.

Special count option

The counter value can increase or decrease. If the start value is bigger than the end value, the counter is decremented.

Figure 55. Loop decrement

```
Loop (mycount; 3; 1)
```

In the countdown example in *Figure 55*, mycount takes values 3, 2, and 1.

The counter can have a specific increment value when a third value is added to the loop instruction, as shown in *Figure 56*:

Figure 56. Loop specific increment

```
Loop (mycount; 1; 6; 2)
```

This example counts with a step of 2. Successive values are 1, 3, and 5. The loop stops at 5 because 7 is higher than 6.

The loop can include another loop. It is mandatory to use a different counter name.

Figure 57. Nested loop

```
Loop (row;4;5)
  Loop (column;3;2)
    Pause ("coord: [row] [column]")
  EndLoop
EndLoop
```

The script provided as nested loop example in *Figure 57* displays: *coord: 4 3*, *coord: 4 2*, *coord: 5 3*, and *coord 5 2*.

Loop script verdict

The loop generates some special lines in the verdict file. The added lines help the user to follow the execution.

The script shown in *Figure 58* generates the verdict shown in *Figure 59*:

Figure 58. Loop script verdict example

```
Loop (FREQ,13,15)

EndLoop
```

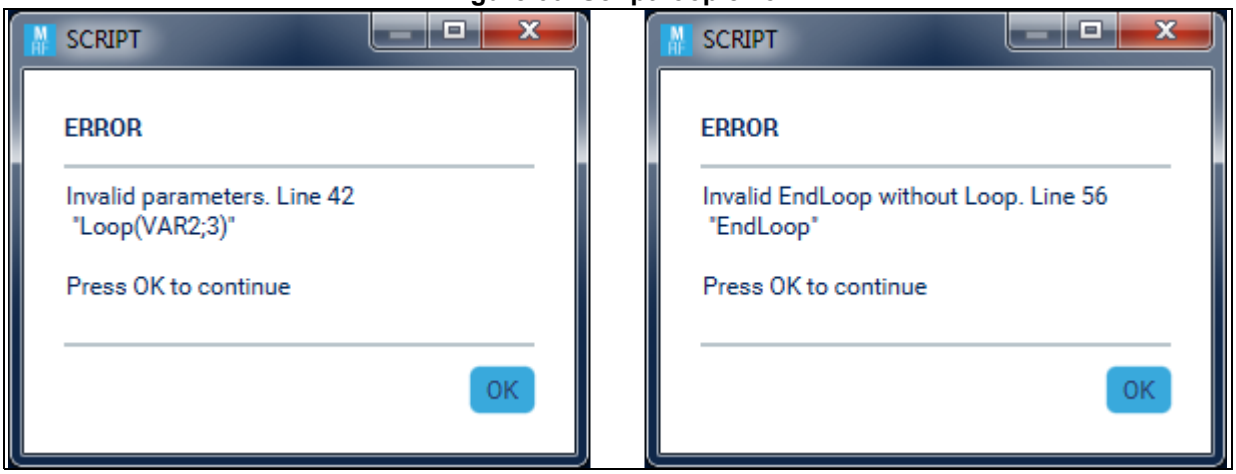
Figure 59. Loop script verdict display

```
-- Loop Start (FREQ=13) --                    | --      |             |
-- Loop (FREQ=14) --                          | --      |             |
-- Loop (FREQ=15) --                          | --      |             |
-- Loop End (FREQ) --                         | --      |             |
```

The beginning and end of the loop are indicated, and the counter value is also inserted in decimal for each iteration.

In case of an error in an instruction, a pop-up warns the user when the script is executed, and the line is skipped. If a Loop instruction is missing or invalid, the EndLoop generates the *Invalid EndLoop without Loop* warning message.

Figure 60. Script loop error



3.5 OTA transfer

3.5.1 OTA presentation

Over-the-air (OTA) transfer executes the transfer of data from a device to a remote device without a cable. The data are applicative data, like user configuration, pictures, music, or firmware. STM32CubeMonitor-RF provides a transfer function from the computer to the remote device over Bluetooth low energy.

In this section, the computer or device sending the data is named *Source device*.

The data are transferred by the source device and the OTA loader to the address requested by the user.

The implementation example does not include security in the transfer process. It is expected that the user changes his loader or application to perform the security verification based on customer requirements.

The OTA process is described in the application note *Over-the-air application and wireless firmware update for STM32WB Series microcontrollers* (AN5247), available on www.st.com. Read these documents for the details of device configuration and OTA procedure. In this user manual, there is only a summary of the procedure, to explain how to use the tool. Read the application note to get detailed information about the target software and the Bluetooth low energy services used.

OTA loader

The OTA loader is the first application that started at boot or reboot. OTA loader checks the boot conditions, and if Flash is empty.

When the bootloader starts in OTA mode, the loader creates an OTA service and some characteristics required to perform the OTA transfer. These attributes are used to perform the transfer.

The loader is fitting in the first 6 sectors of the Flash memory, so the block at address *0x6000* is free and used to upload the user data.

Table 1. OTA loader address table

Flash address:	Flash content:
+0x0000	OTA bootloader
+0x1000	OTA bootloader
+0x…	OTA bootloader...
+0x6000	Free for user data
+0x7000	User application
+0x8000	User application
+0x….	User application...

- In the STM32WBxx sample code, the binary is stored at address *0x7000*, and the bootloader starts at this address after upload.

3.5.2 OTA procedure

The OTA procedure occurs between one source device and the target device. The process is based on operations:

1. Activate the OTA mode on the target device.
2. Connect in OTA mode and transfer data.

Activation of OTA mode

The computer sends an indication to the target device to reboot in OTA mode, with the download information. The target restarts in OTA mode and erases the flash area required for the transfer.

Connect in OTA mode and transfer data

The source device first connects to the OTA loader and discovers the details of service and characteristics to be able to transfer the data. Then the sequence is:

1. Configure the target device to send an indication to the source device.
2. Write in the target device the command to initiate the procedure, with the exact storage address.
3. Write each block of data. Depending on optimized or not MTU size, the blocks are 20- or 248-byte long, and the binary must be transferred in many blocks.
4. At the end of the last block, write the confirmation that all blocks have been sent.
5. The source device waits for the reboot confirmation from the target.

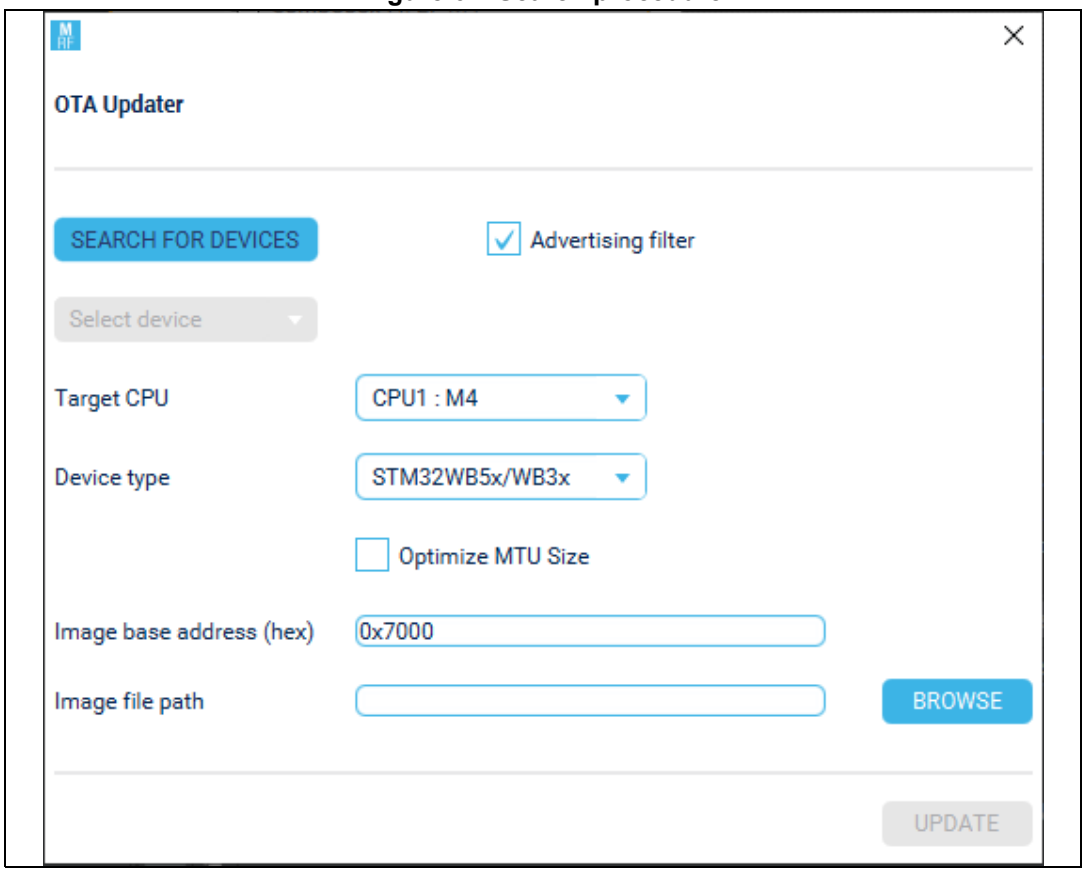
3.5.3 Use the tool to perform an OTA update

The OTA function is available in the device menu in the menu bar. Click on the device and then click on OTA updater.

Search procedure

The first operation is to find the target device. The tool needs to perform a scan of Bluetooth low energy devices and list all the devices with OTA capabilities.

Figure 61. Search procedure



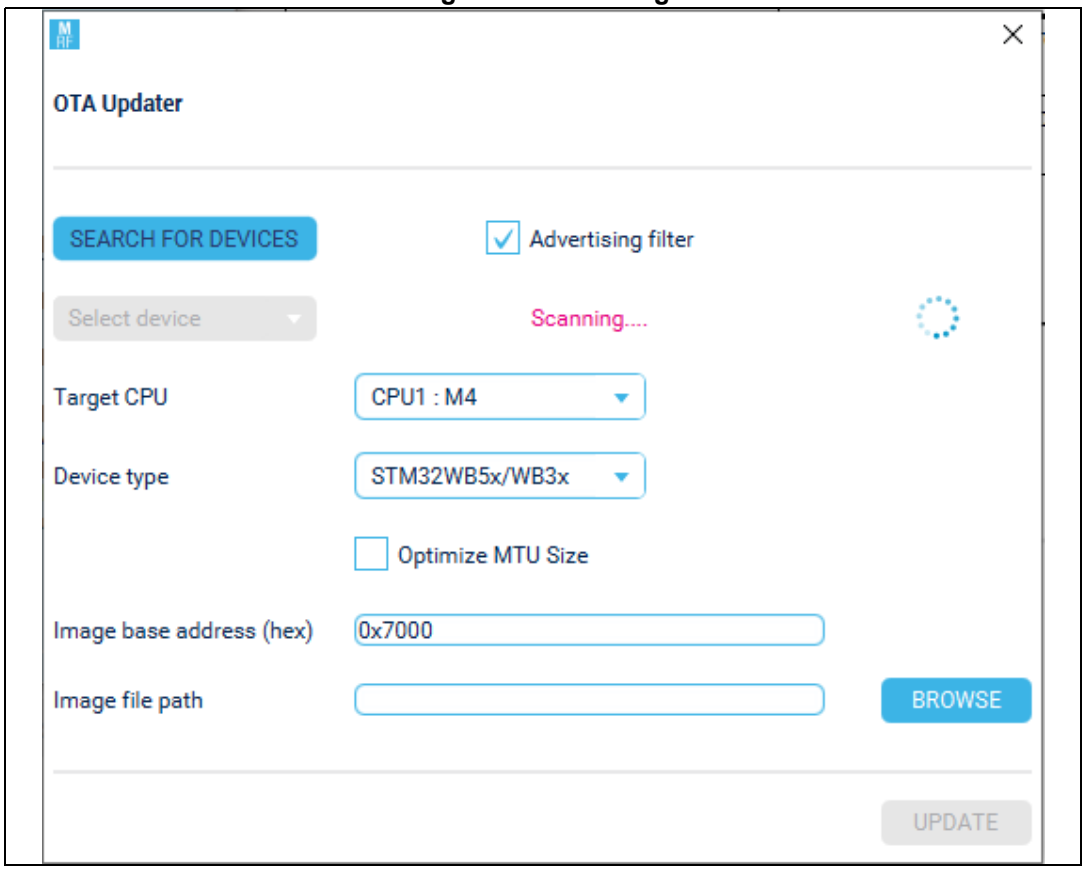
The tool provides an advertising filter to refine the search procedure with an advertising message.

Table 2. Search filtering

Filter	Search method	Comment
No filter	Scans all Bluetooth low energy devices and provides the list.	Some devices listed are not compatible with OTA.
Advertising filter	Scans all Bluetooth low energy devices and provides a list of devices with ST OTA information.	Gives only the list of compatible devices.

To start the search, click on the *SEARCH FOR DEVICES* button.

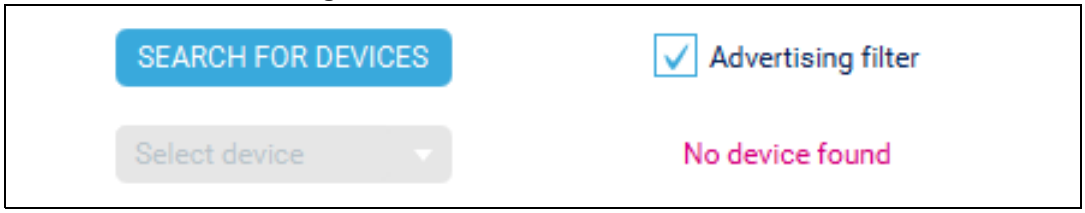
Figure 62. Scanning



The search procedure starts.

If no target device is found, the tool indicates *No device found*.

Figure 63. No device found



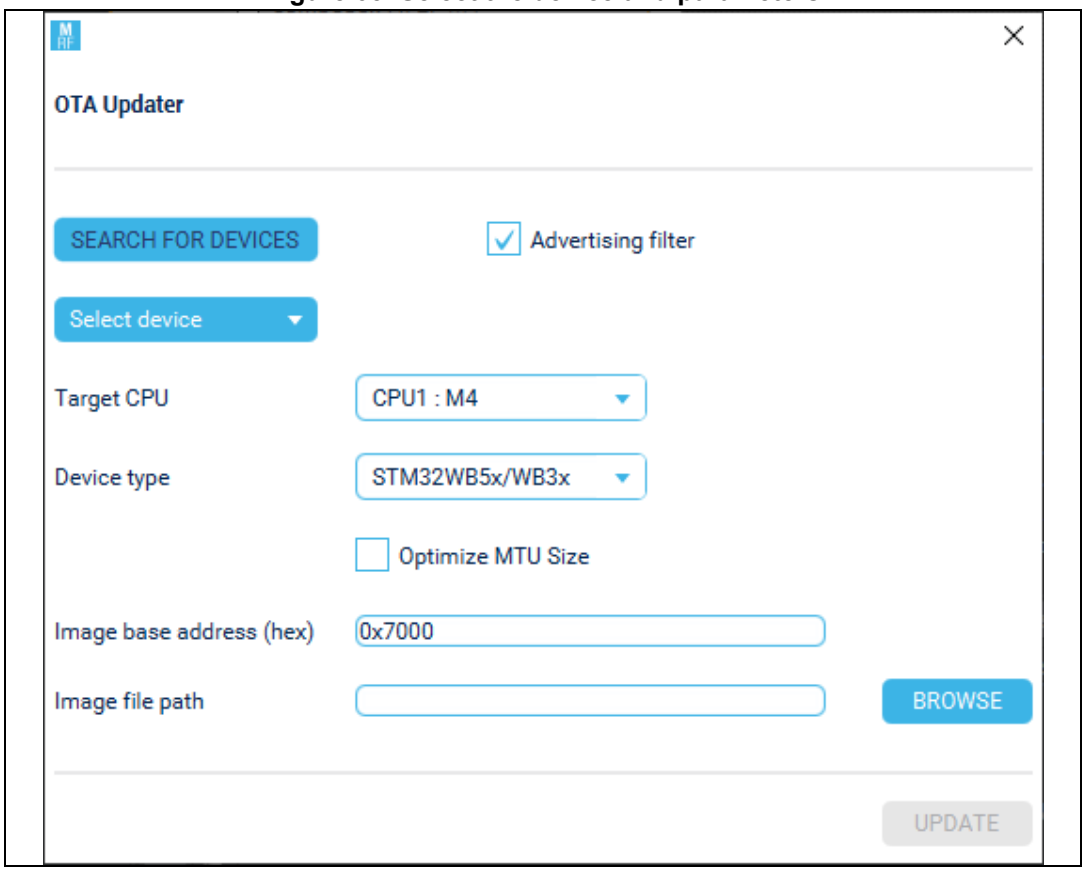
If a candidate device is found, the select device box changes to blue.

Figure 64. Device found

Select the device and parameters

After the search procedure, if one or more devices are found, the user selects the device with the picklist *Select device*.

Figure 65. Select the device and parameters



The picklist displays the list of boards found:

- For a device with Bluetooth low energy characteristics:
  Bluetooth low energy address - Device name - OTA enabled
- For a device already in OTA mode:
  Bluetooth low energy address - Device name - OTA loader

Select the firmware target:

- For user data or user application firmware, select the *CPU1: M4*
- For Wireless stack, select the *CPU2: M0+*

Select the device type:

- STM32WB5x/WB3x product lines
- STM32WB1x product line. Note that for this device type only Target CPU1: M4 can be updated.

Optimize MTU size allows the user to increase the ATT maximum transmission unit (MTU) from 20 to 248 bytes.

The image base address is the place where the binary file must be stored on the target device. It is a hexadecimal value and must be a multiple of *0x1000* for STM32WB5x/WB3x or *0x2000* for STM32WB1x to match with the Flash sector. For the wireless stack, the address is the temporary location in the CPU1 User part area.

The image file path is the binary file to load. Enter the path in the box, or use the BROWSE button to select the file to download.

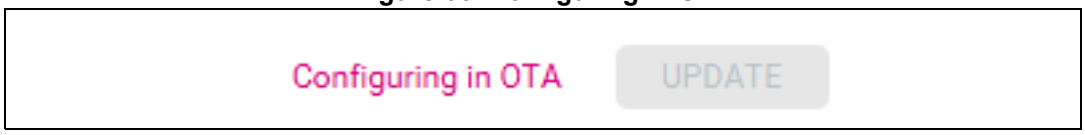
The configuration is finished and the software is ready to start the update procedure.

Flashing the remote device

Press the *UPDATE* button to start flashing the target device.

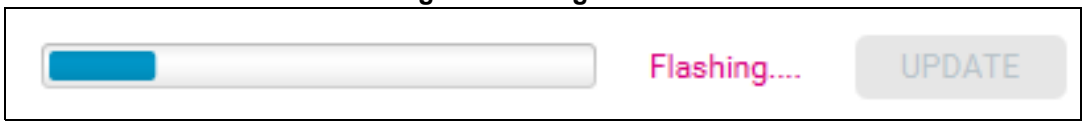
1. First step: if the selected device has an OTA characteristic, the tool first restarts the device in OTA bootloader mode. The indication *Configuring in OTA* is displayed

Figure 66. Configuring in OTA



2. Second step: the transfer process to the OTA bootloader is performed. The data are transferred by blocks of 20 bytes. To avoid the overload of log windows, the log information related to block transfer is not displayed. Only the flow control event and errors are displayed.
3. The progress is indicated by a progress bar.

Figure 67. Progress bar



At the end of the update process:

4. The target device reboots.
5. The user closes the OTA panel or starts a new search to flash another device.

3.6 Beacon

3.6.1 Beacon presentation

A beacon is an active device discoverable by other devices.

The beacon device only sends information by advertisement and does not receive any data.

The data shared by the beacon are very small. They are received by a connected device and the application on the device is notified of beacon presence. The application uses the cloud to get more information and act accordingly.

Figure 68. Beacon presentation

When an application is informed of beacon proximity, it uses the beacon identification to request the web server more information about the beacon. The application gets information related to the geographical position of the beacon or action to perform, like displaying commercial ads or start an interactive application.

Figure 69. Beacon usage

Many organizations have created beacons. The specifications from Apple® and Google® are frequently used:

- iBeacon: this is Apple format. The beacon broadcasts a fixed content, to easily identify the beacon.
- Eddystone UID: this is defined by Google. The beacon transmits fixed content (UID), which is a unique ID, referenced in the Google database to interact with applications.
- Eddystone URL: another Google format, provide a short URL, to use for the *Physical web*.
- Eddystone TLM: an additional beacon advertising information providing beacon information (battery status, temperature).
- Eddystone EID: like UID, but broadcasts encrypted data to provide better security.

3.6.2 Beacon configuration methods:

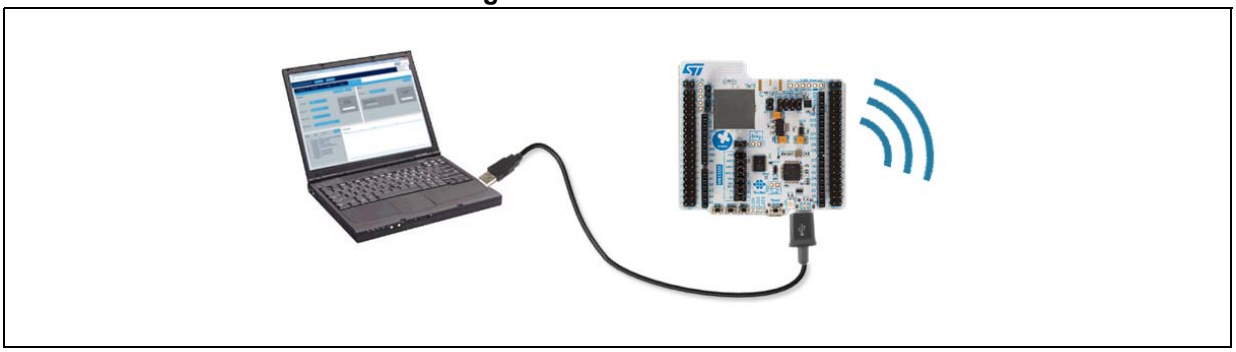
STM32CubeMonitor-RF is used to generate and configure beacons. Different methods have been defined to accommodate the user's needs. This chapter describes the different methods supported.

Online beacon

In online mode, the tool is directly configuring the main device in a beacon. The tool sends ACI commands to configure the boards in advertising mode and configure the content of the advertising packet. The main device acts as a beacon until turned off.

The main advantage of this method is to quickly configure a beacon with a board in transparent mode. The drawback is that the configuration is lost when the board is reset or powered off.

Figure 70. Online beacon

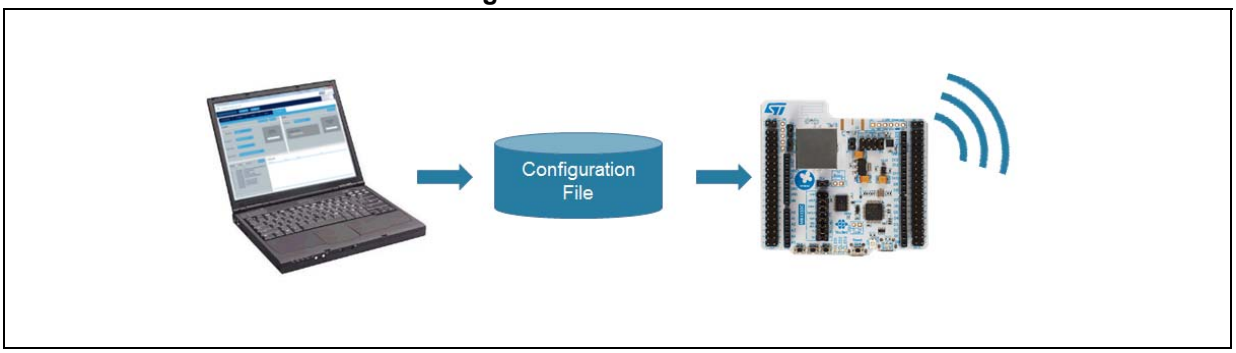


Offline beacon

The offline beacon mode is used to prepare the configuration of a board not directly connected to the STM32CubeRF-Monitor. The parameters to configure the beacon are stored in a data file. The file is used to configure a target board running the beacon example firmware. The file must be stored in the target Flash at address *0x6000*. The beacon firmware reads the data and configures the advertising block accordingly. Details of the configuration file are described in *Table 4: Beacon configuration format*.

The interest of the method is to have an independent beacon, which is useful if the user needs many beacon boards at the same time. It is possible to keep many configuration files to change configuration quickly. The drawback is that the configuration file must be transferred manually in the target device so it is less flexible than the *Online* mode.

Figure 71. Offline beacon

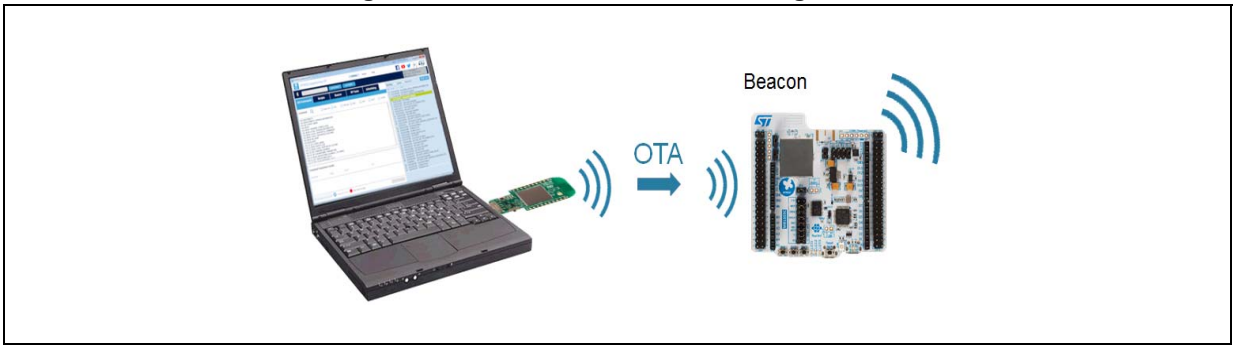


Beacon over-the-air configuration

This method is like Offline mode, but the tool uses the OTA procedure to transfer the configuration file directly into the target Flash. The target board with a beacon demo

firmware directly restarts after OTA with the updated parameters. (OTA needs to be enabled). The main device is used to transfer the file to the target device.

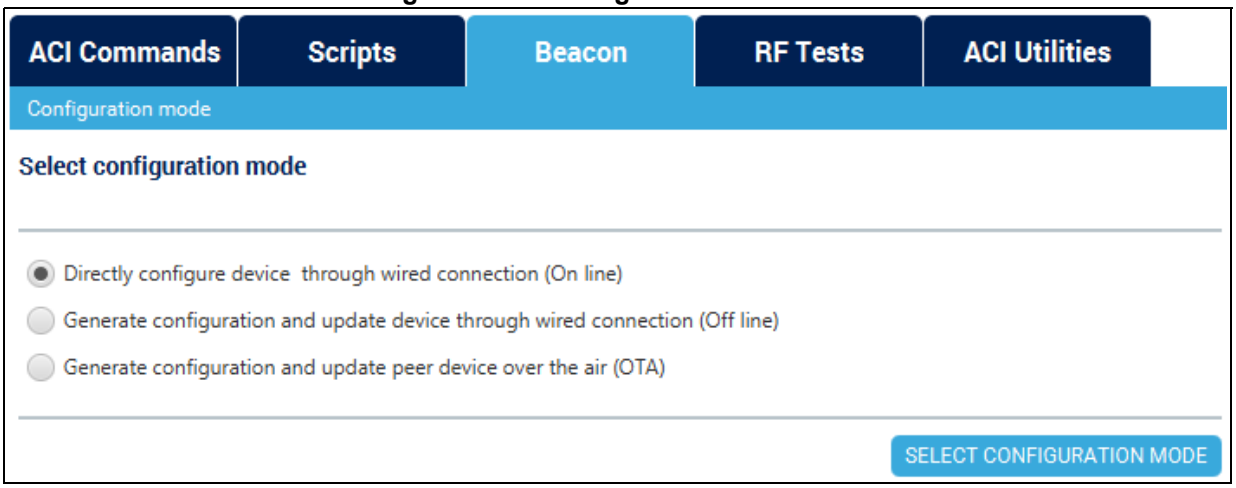
Figure 72. Beacon over-the-air configuration



Selecting the beacon mode

The selection of the configuration mode is the first action to prepare the beacon. The user must select the mode when he selects the beacon tab.

Figure 73. Selecting the beacon mode



Select one of the three bullets and click on *SELECT CONFIGURATION MODE*.

3.6.3 Configuration of the beacon with STM32CubeMonitor-RF

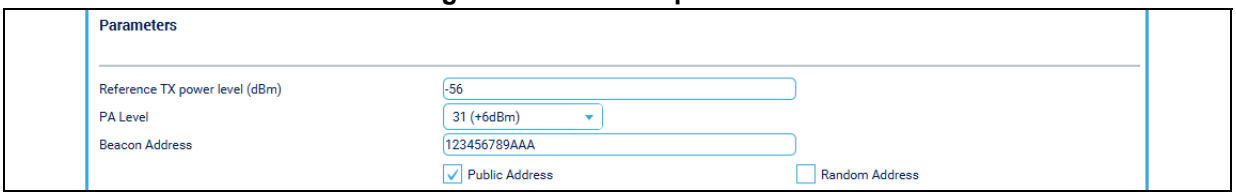
To configure the beacon:

1. Select the configuration method.
2. Fill the beacon parameters, some are common for all beacons, others are specific for the beacon type.
3. Generate/transfer the configuration. Additional information may be required according to the configuration method.

Common parameters

Some beacon parameters are common for all kinds of beacons. The common parameters are at the top of the beacon panel:

Figure 74. Common parameters



The first parameter is the *Reference TX power level*, and the second parameter is the real *TX power level*.

To save batteries, the power level of the beacon may be lowered, reducing consumption and visibility. Using high power extends the range of visibility but drains more power. The power level needs to be defined by the user based on power source and beacon purpose.

The device detecting the beacon needs to estimate if the beacon is close or far. Unfortunately, the received power level is not enough to estimate the real distance:

- Some beacons may transmit with high power, while others are using low power.
- The design of the beacon antenna may be efficient.

The reference power information is added to help determine the distance. This is the power level received at one meter from the beacon. The application uses this value and the received strength to estimate the distance, independently of the real TX power used and the beacon characteristics.

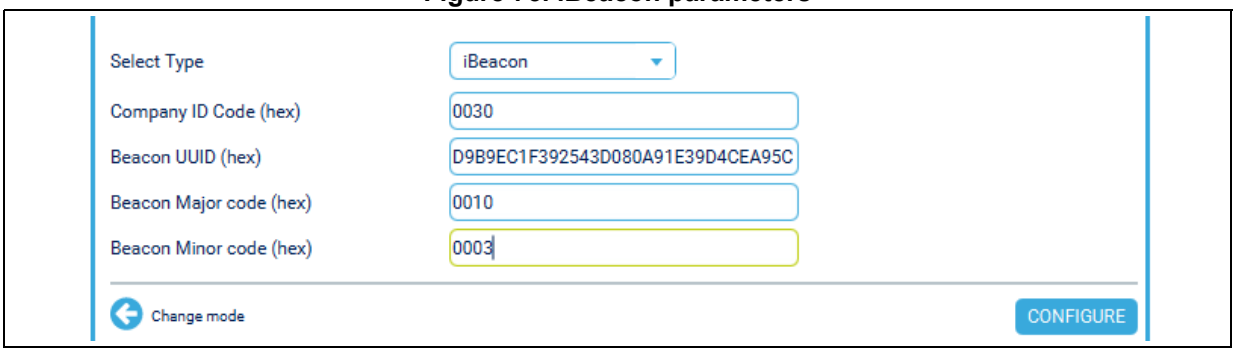
The easiest solution to fill this parameter is to configure a beacon with the required Tx level, and then to measure the received level at 1 meter. Then beacon is reconfigured with the value measured at 1 meter in the *Reference TX power level* field.

The second set of parameters is the beacon address. There are 3 possibilities:

- Set the address in the box and tick the *public address*. The address entered is used.
- Tick the *random address* checkbox. A random address is used.
- If nothing is selected, the board default public address is used.

iBeacon parameters

Figure 75. iBeacon parameters



First, select the type: iBeacon (default choice)

The user must check the Apple website for information about iBeacon structure and the condition to use iBeacon for his project: https://developer.apple.com/ibeacon/

More information is also available at https://en.wikipedia.org/wiki/IBeacon

The company code is a value based on Bluetooth low energy SIG group assigned values. For iBeacon, the Apple value is used: *0x004C*. The assigned values are available on the SIG website: https://www.bluetooth.com/specifications/assigned-numbers/company-identifiers

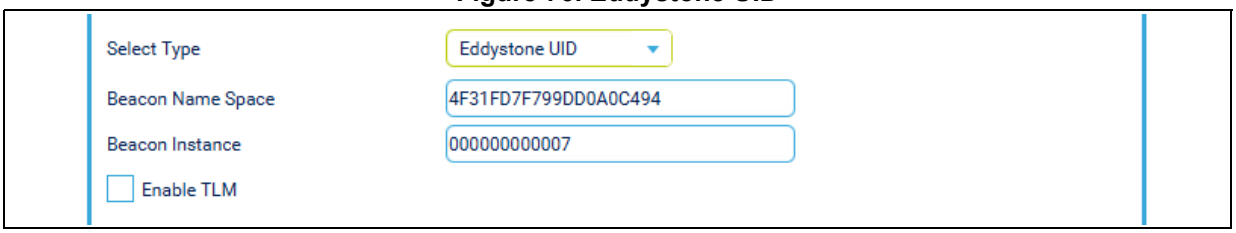
The beacon UUID is the unique identifier for a group of beacons. Apple explains how to define the identifier in the document *Getting started with iBeacon* available at https://developer.apple.com/ibeacon/Getting-Started-with-iBeacon.pdf.

The major and minor codes are defined by the user to identify logically different beacons sharing the same UUID.

When all parameters are updated, click on *CONFIGURE*. The data are ready for transfer (Refer to *Transfer the configuration*).

Eddystone UID parameters

Figure 76. Eddystone UID



The Eddystone UID parameters are the beacon UID, a 16-byte identifier, formed by:

- NameSpace, 10 bytes. Used to group some beacons in a logical pool. The way to generate the value is described by Google, refer to https://github.com/google/eddystone/tree/master/eddystone-uid
- Beacon instance, 6 bytes. Give a unique id inside the pool.

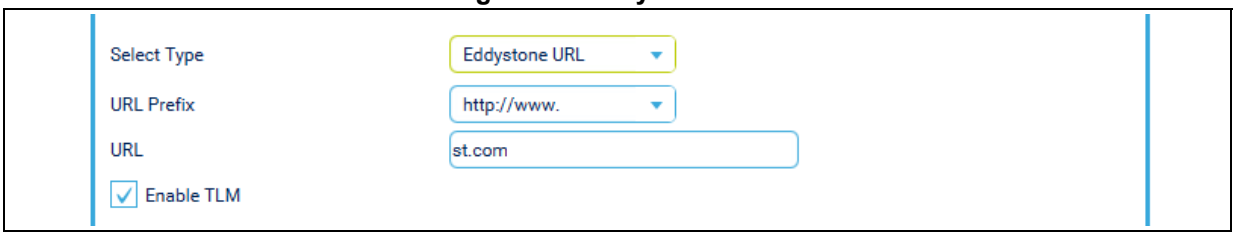
When a beacon is discovered on a smartphone, the UID value is not directly usable by the phone application. Google offers a cloud service to associate one or more data with a beacon. The smartphone application retrieves this information to perform the required actions.

The last option is the Enable TLM tick box. When TLM is used, the beacon interleaves some status information inside the normal beacon advertisement. The TLM frame has information about battery level, temperature, the time beacon is on, and the number of frames transmitted. The TLM information is not known by the tool. So, it must be managed directly by the firmware. Consequently:

- The TLM option is not used for Online configuration mode.
- For Offline and OTA modes, a bit is set in the configuration file (Refer to *Appendix A*).

Eddystone URL parameters

Figure 77. Eddystone URL



The Eddystone URL format is just sending a URL in the advertising message. To optimize space, the start and end of the URL may be compressed.

1. Select the URL prefix: the prefix is encoded in 1 byte in the advertising.
2. Fill the rest of the URL in the URL box, without prefix. The URL is parsed, and if the end of the URL is encodable, the tool encodes it. Long URL does not work, it is advised to use URL short service to get a short URL.

The TLM option is the same as the UID beacon.

Transfer the configuration

The transfer depends on the selected configuration mode.

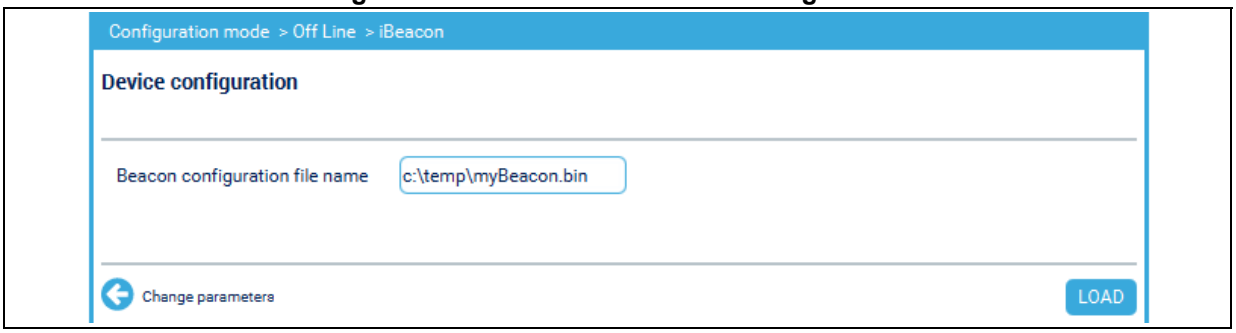
1. Online mode transfer configuration

Figure 78. Online mode transfer configuration

No extra parameters are required: just click on *LOAD* and the main device is initialized and configured in the beacon.

2. Offline mode transfer configuration

Figure 79. Offline mode transfer configuration

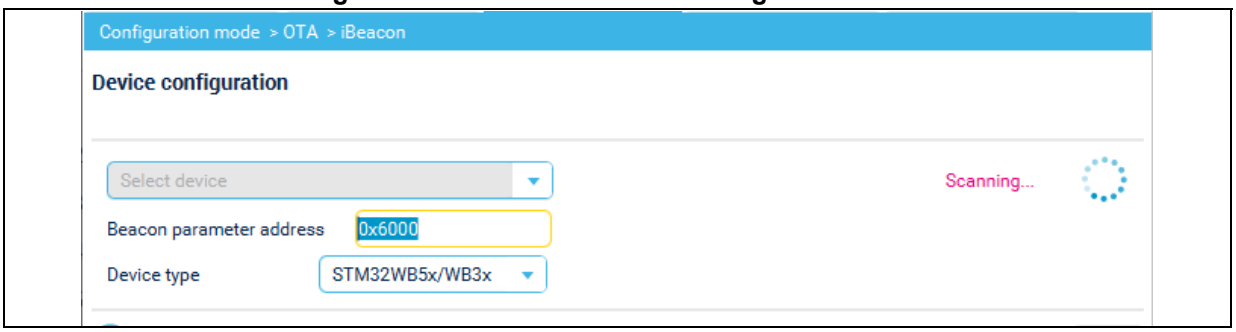


Indicate first the name of the file to create, including the path. If no path is provided, the file is stored in the tool directory. The file is then copied to the target device using a Flash programmer, or any other tool.

3. OTA mode scan transfer configuration:

   a) Scan
   As soon as the page is displayed, the tool asks the main device to search OTA-capable devices in the area. *Scanning* is displayed in the windows.

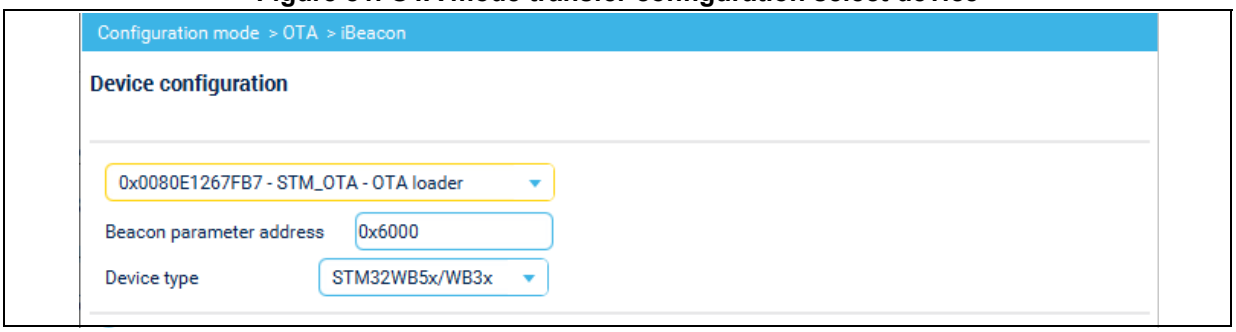
Figure 80. OTA mode transfer configuration scan



   b) Select device
   When the device is found, the list is updated, and the user selects the device to configure. The address to store the beacon data is *0x6000* in ST example firmware. The value is editable if another address is used.

   c) Select device type
   Choose the device type that the user wants to flash, either STM32WB5x/WB3x or STM32WB1x product line.

Figure 81. OTA mode transfer configuration select device

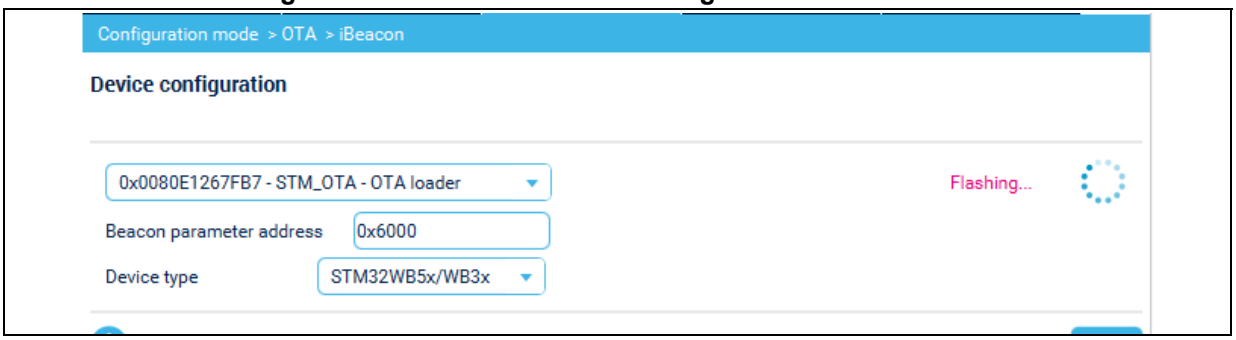


   d) Load the file
   Click on *LOAD* and the tool flashes the binary data.

Most of the time, beacons are not connectible and are moved in the OTA loader by the user's action. If the beacon is in OTA loader mode, data are transmitted directly.

If the device is OTA enabled, the tool changes it in OTA loader mode first and then transfers the data.

Figure 82. OTA mode transfer configuration transmission

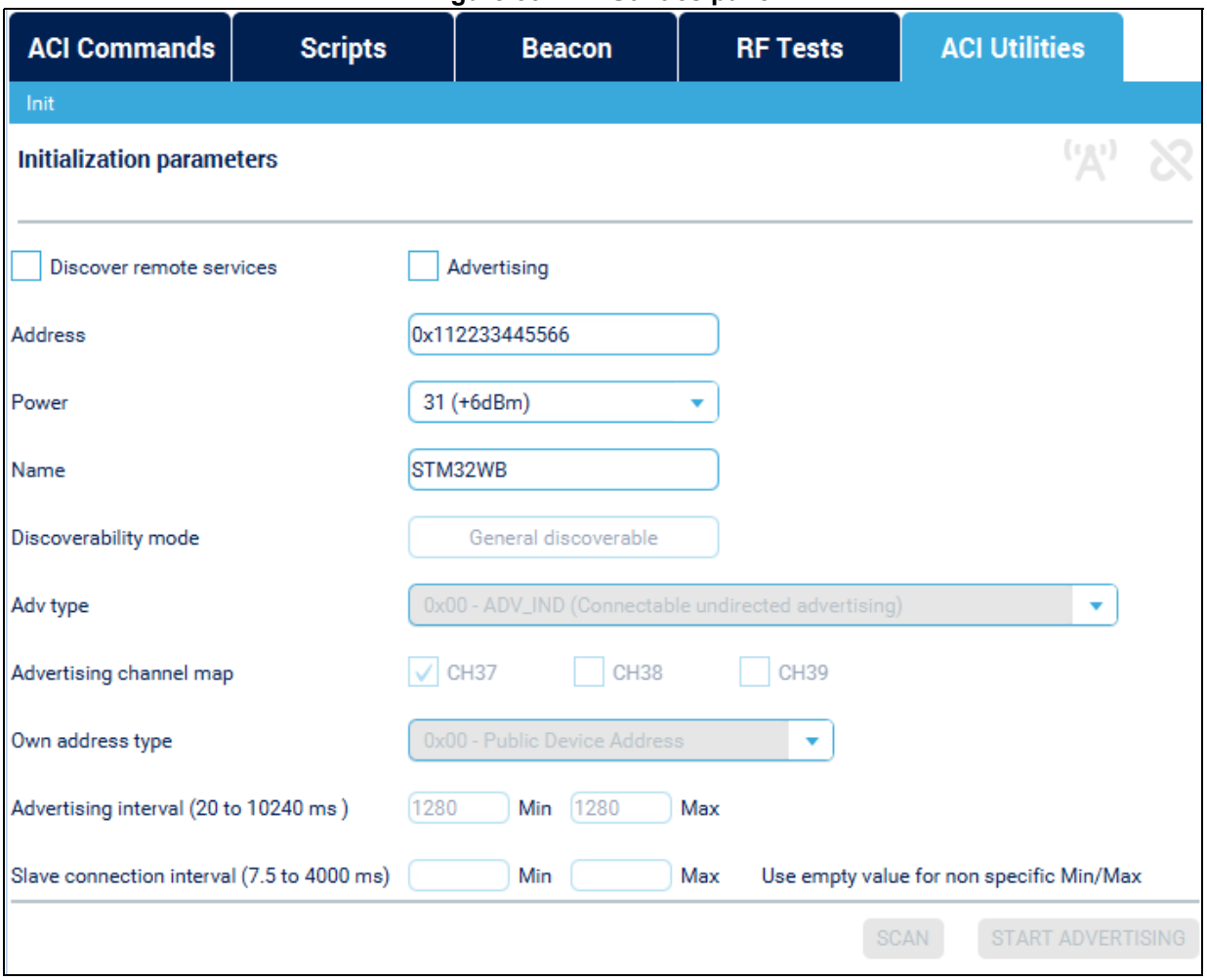


After transfer, the application is restarted, and the beacon is configured.

3.7 ACI Utilities

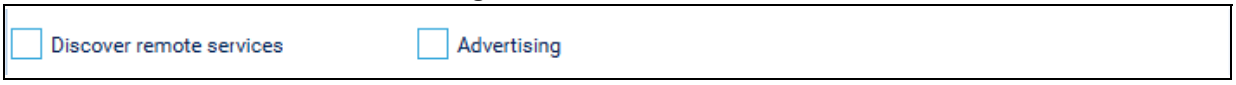
The *ACI Utilities* panel is used to configure the device to perform either the advertising signal or to discover remote devices and explore its services and characteristics.

Figure 83. ACI Utilities panel



The first action is to select to discover remote services, to manage advertising, or both, by clicking the appropriate checkbox.

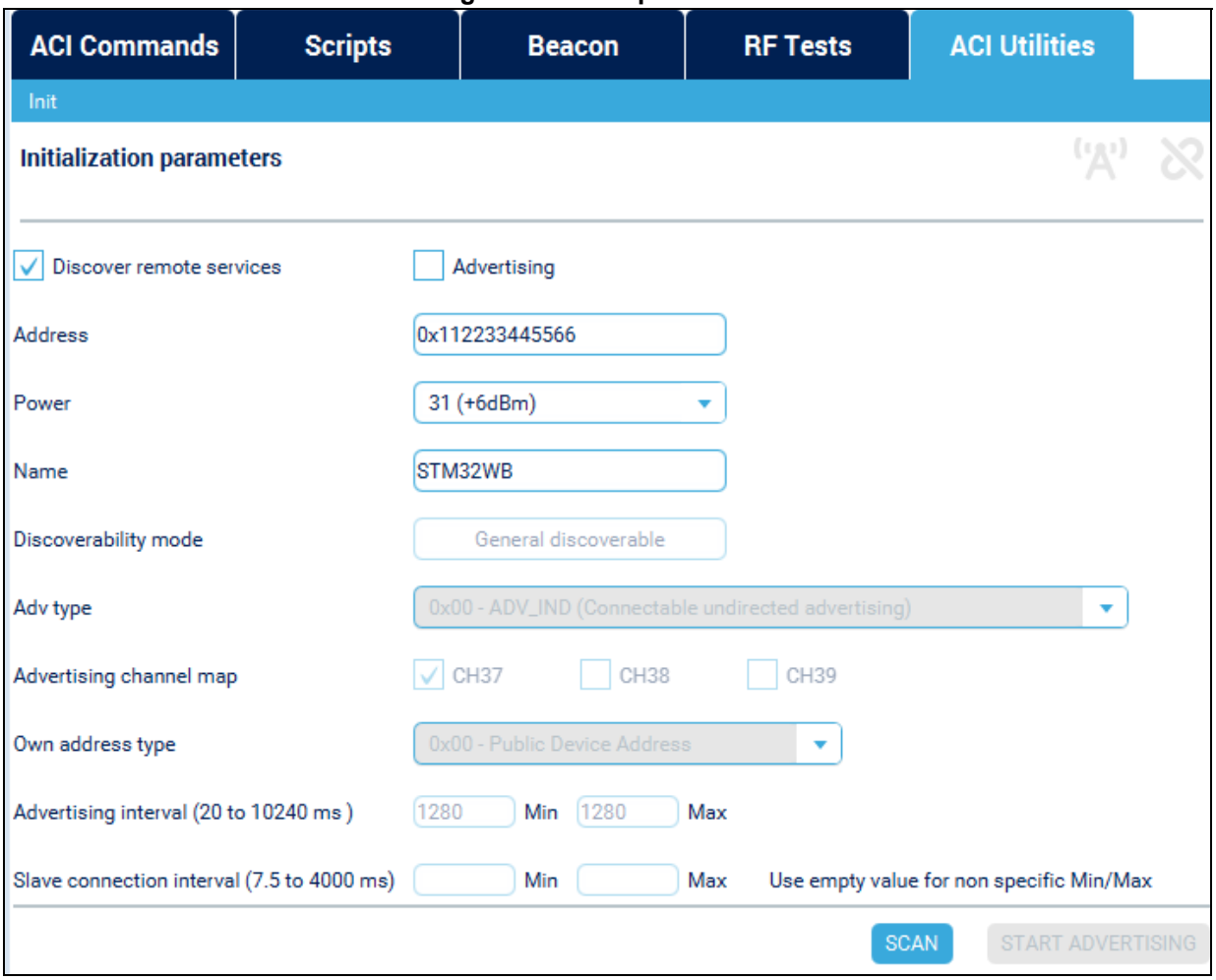
Figure 84. Select checkbox



3.7.1 Remote services discovering

The remote services discovery performs a scan of the remote devices in the area.

Figure 85. Scan parameters

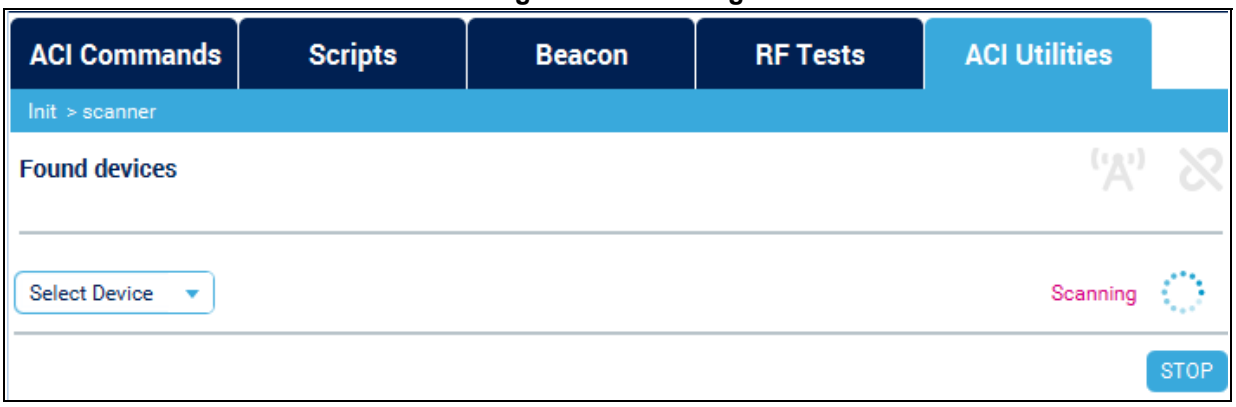


To perform a scan of the available devices:

1. Enter the device address
2. Select the power level with the picklist
3. Enter the device name
4. Click on the *SCAN* button to start the discovery

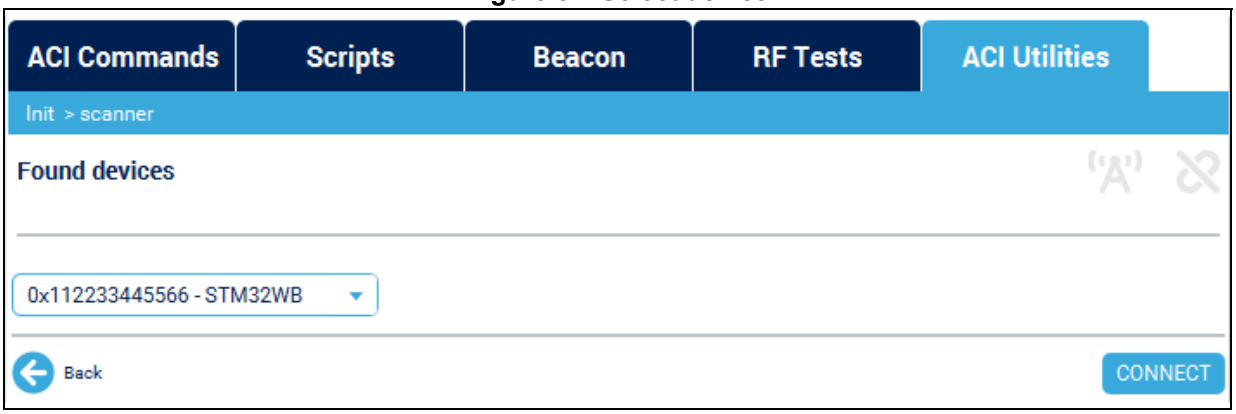
The search procedure starts, and it is possible to stop it using the *STOP* button.

Figure 86. Scanning



If no remote device is found, the tool indicates *No device found*. Otherwise, the user chooses one of the devices found in the *Select Device* box.

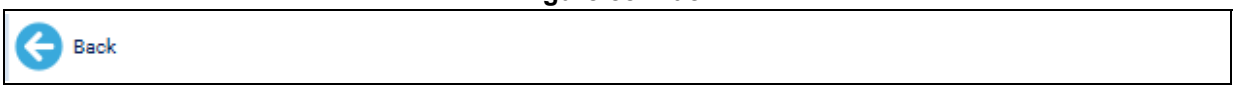
Figure 87. Select device



At this stage, the user performs another scan procedure upon request.

- Click on the back button

Figure 88. Back

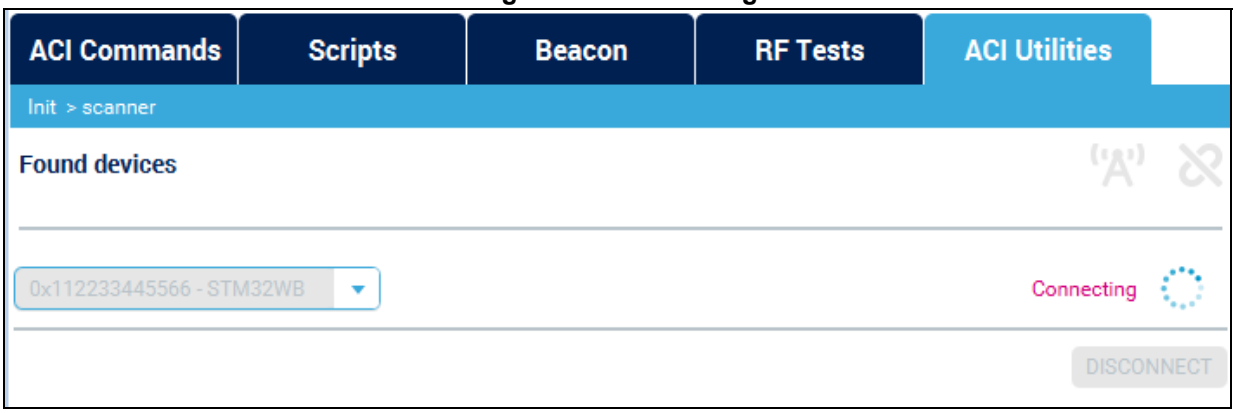


- Click on the *Init* in the top bar

Figure 89. Init

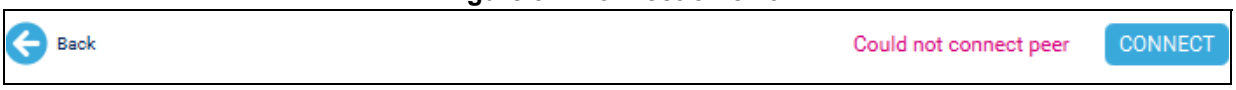
Or connect to the selected remote device, by clicking on the *CONNECT* button.

Figure 90. Connecting



If the connection fails, an error is displayed.

Figure 91. Connection error



Once connected, the connect icon appears in blue and the list of available services is proposed.

Figure 92. Connected icon

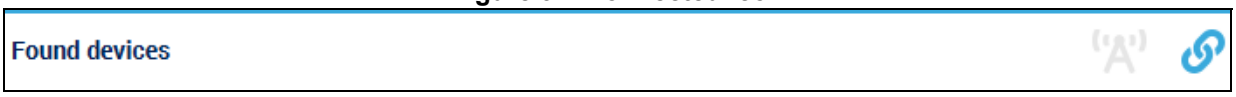
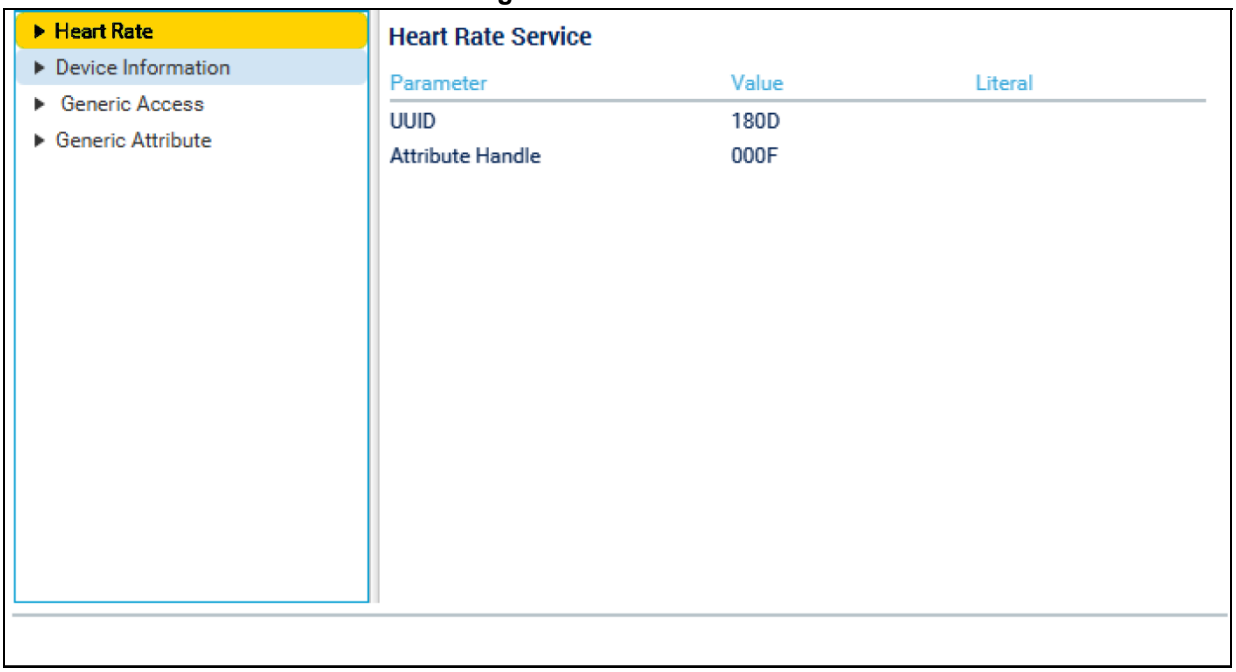
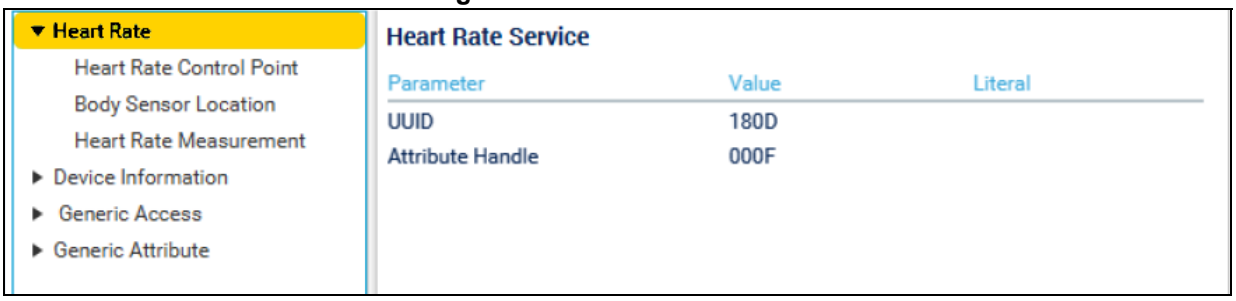


Figure 93. Services list



When the user selects a service, its details are displayed. Clicking on the arrow displays the characteristics linked to the above service.

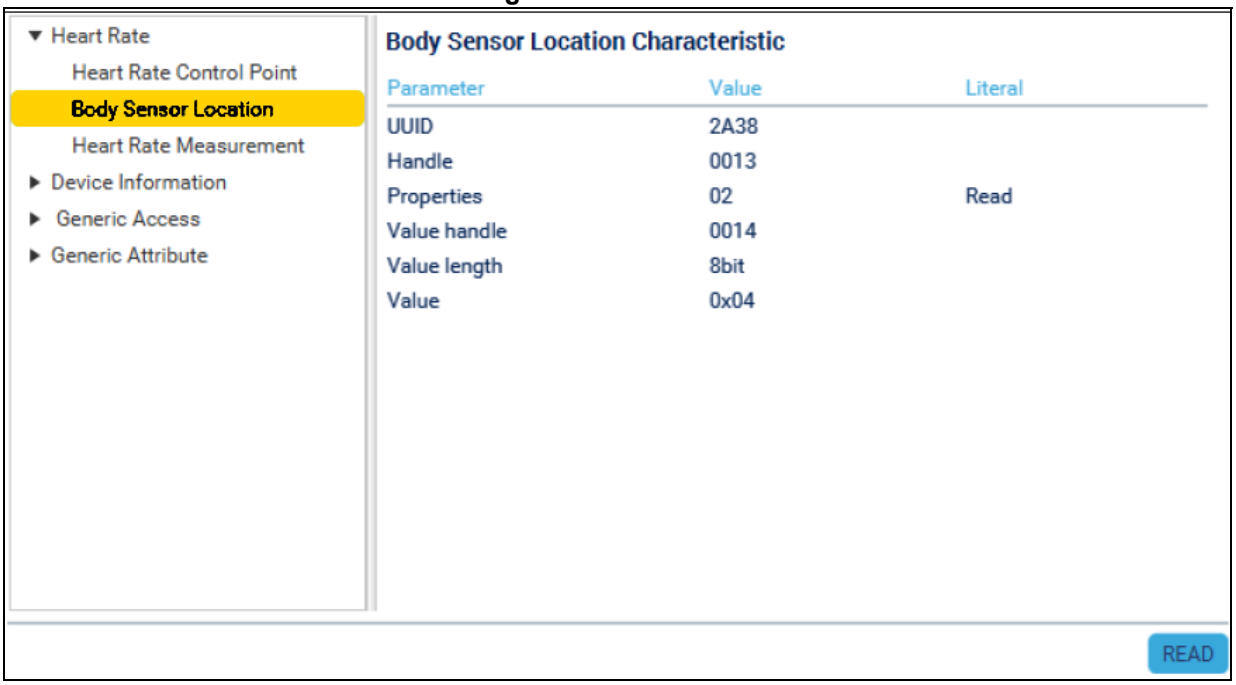
Figure 94. Characteristics list



The user can select a parameter and, depending on each of them, can read or write a value and be notified of value changed. Note that read and write long characteristics are not supported, neither authenticated signed write.

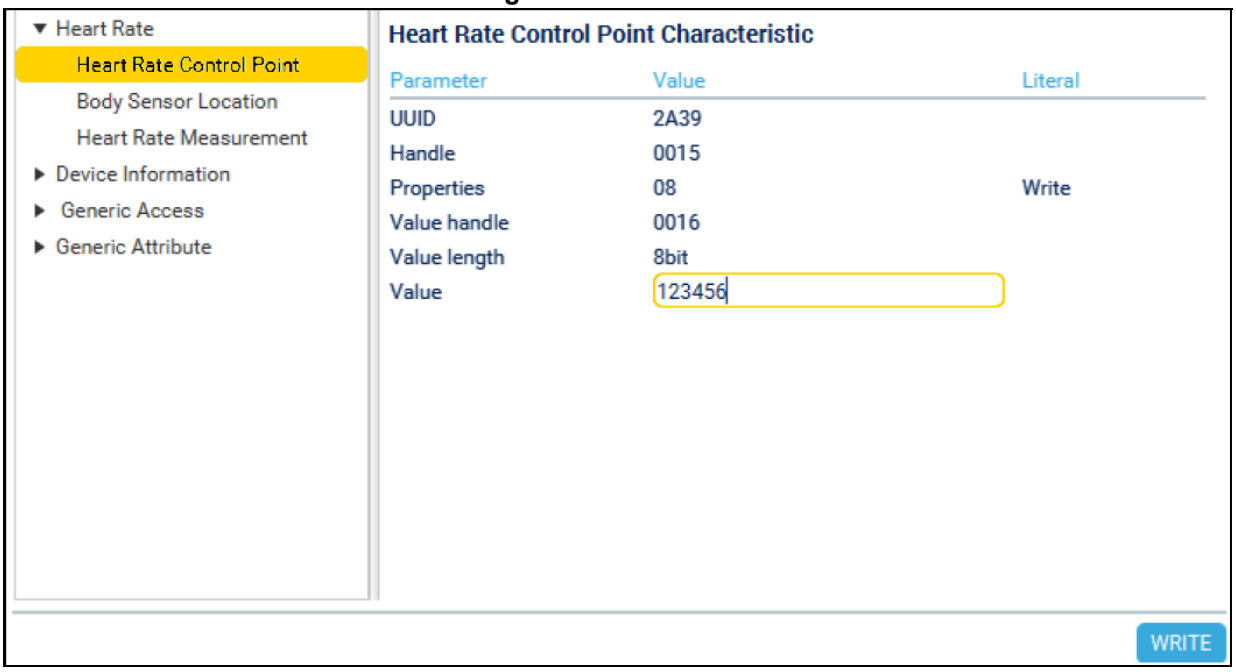
To read a value, the user clicks on the *READ* button.

Figure 95. Read value



To write a value, the user enters the new value and click on the *WRITE* button.

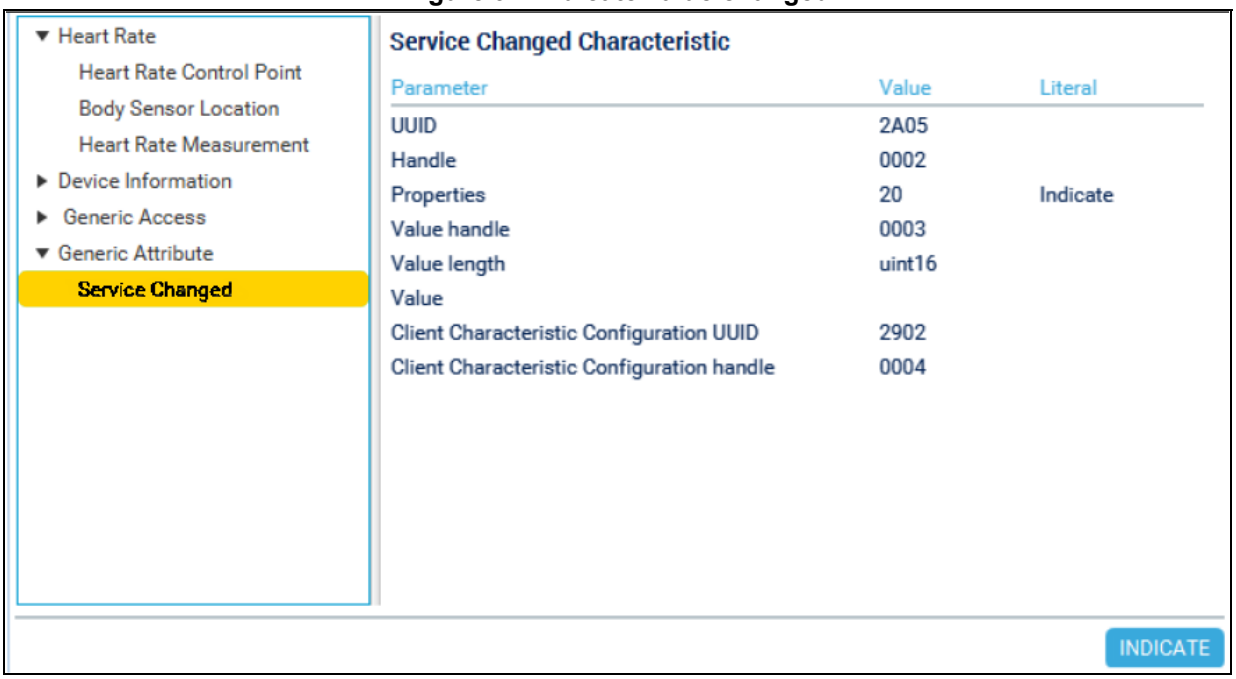
Figure 96. Write value



There are two ways to be informed on a value change, either via the indicated method or via a notification depending on the method property supported by the remote device.

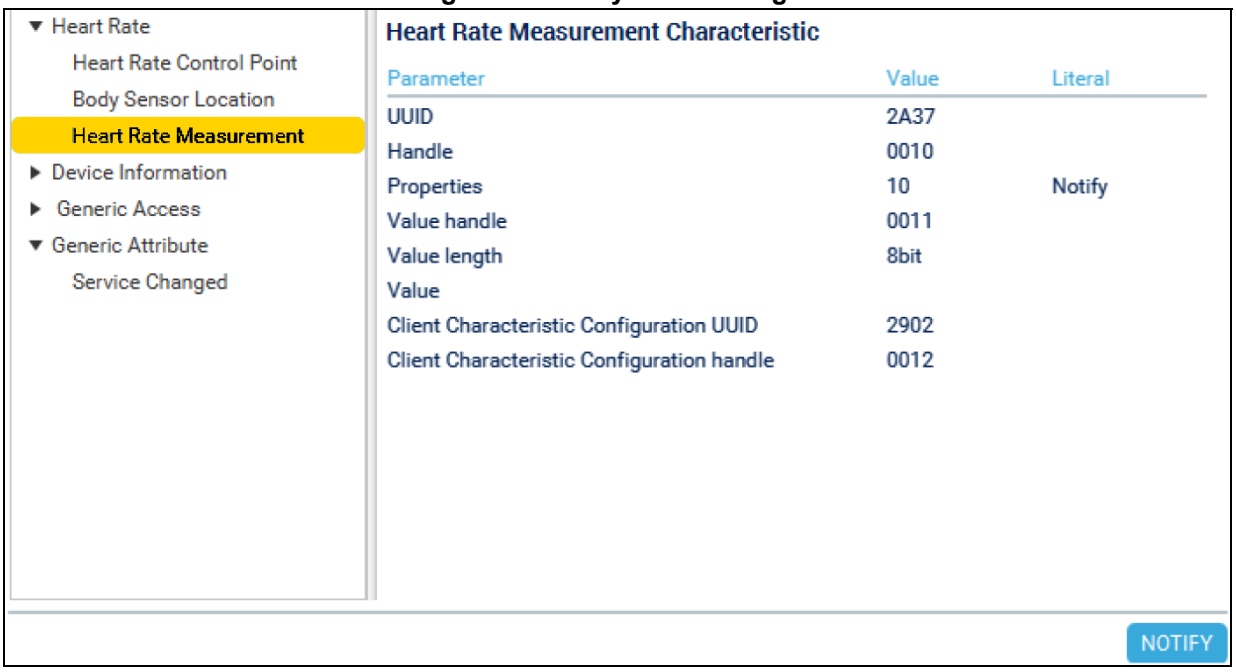
To receive an indication upon value change, the user can click on the *INDICATE* button.

Figure 97. Indicate value changed



To receive a notification upon value change, the user can click on the *NOTIFY* button.

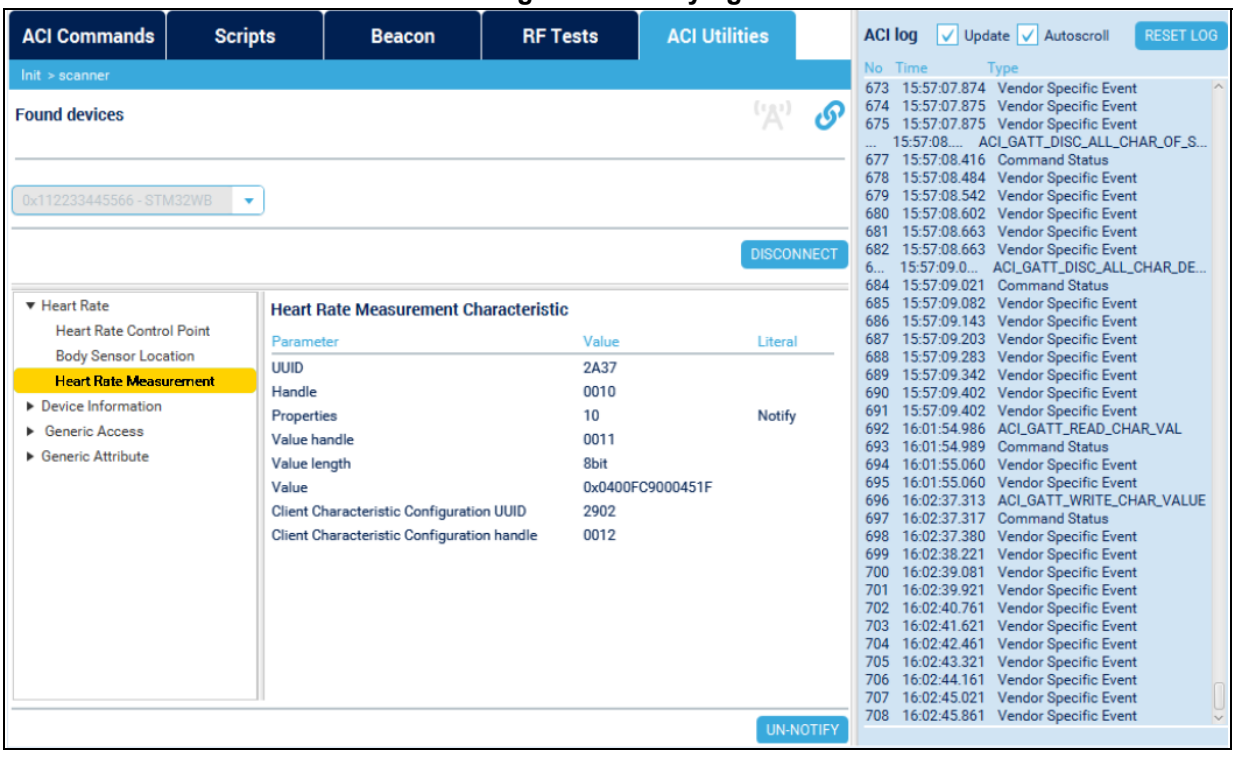
Figure 98. Notify value changed



Upon each change, a notification (resp. indication) is received and the new value is displayed. The user can be informed on multiple characteristics value changes at the same time. To stop notification (resp. indication), the user can click on the *UN-NOTIFY* button (resp. *UN-INDICATE*).

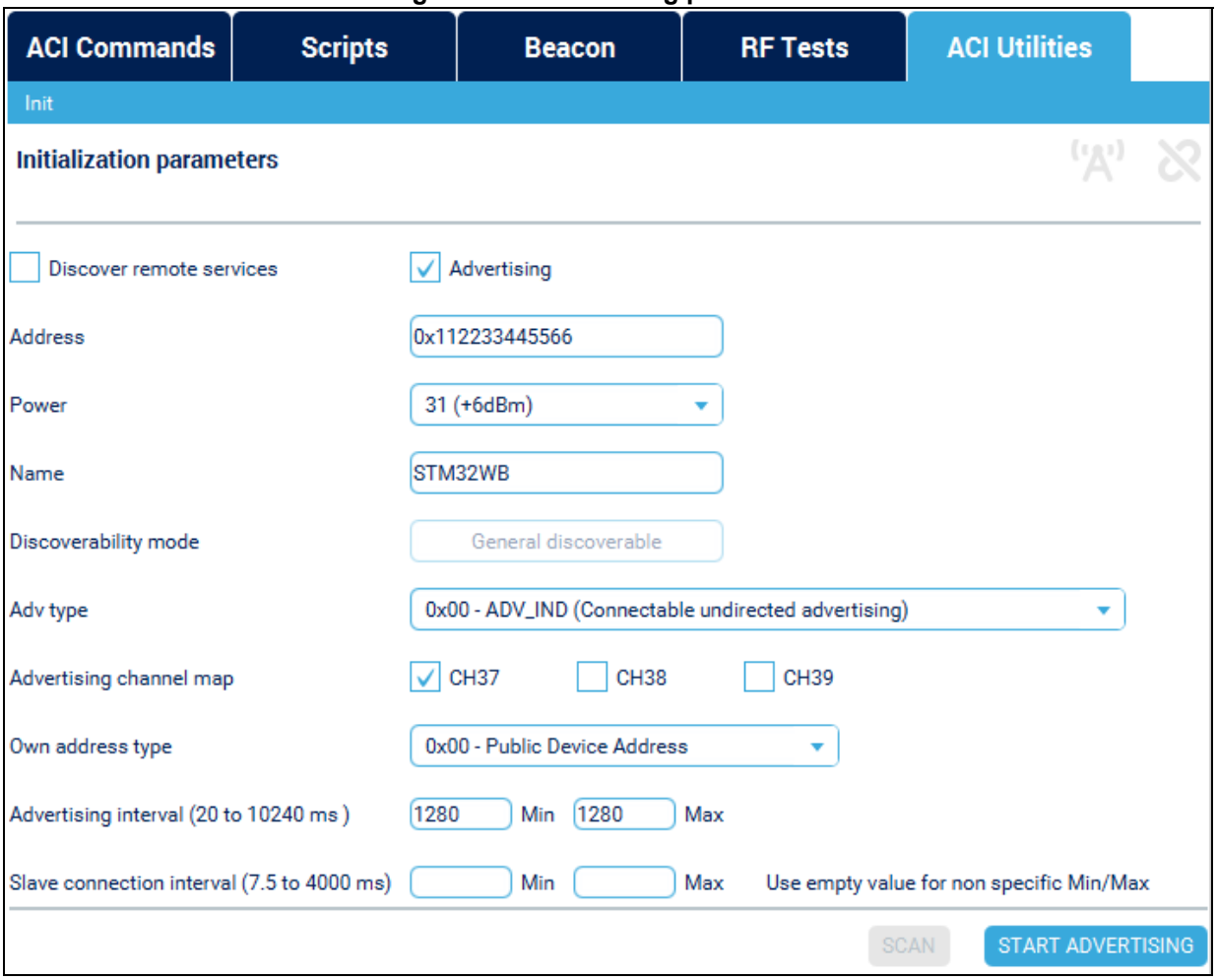
On disconnection, all registered notifications are removed.

Figure 99. Notifying



3.7.2 Advertising

Figure 100. Advertising parameters

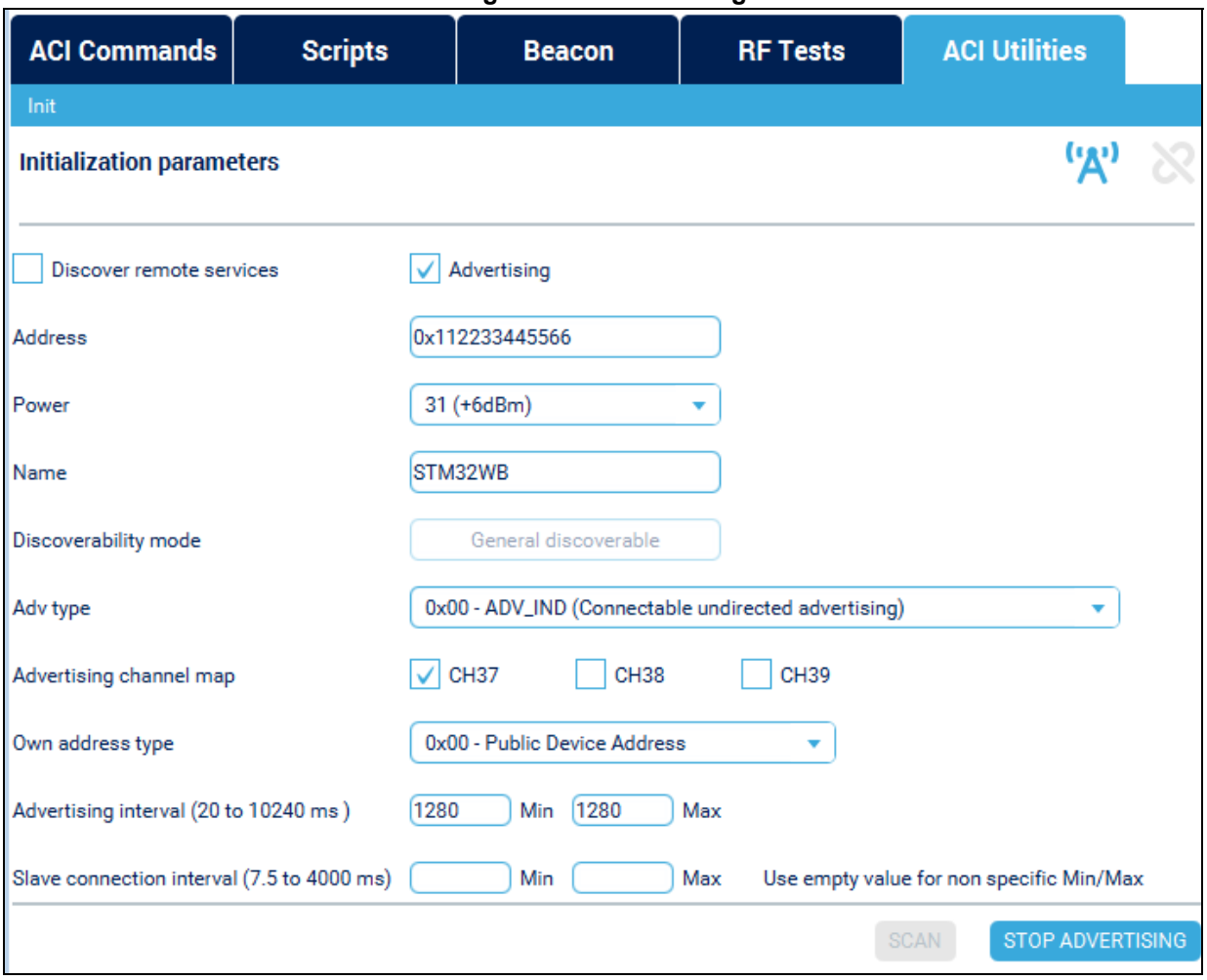


To activate the advertising mode:

1. Enter the device address
2. Select the power level with the picklist
3. Enter the device name
4. Select the advertising type with the picklist
5. Select at least one channel from 37, 38, and 39.
6. Enter the advertising interval
7. Enter an optional slave connection interval
8. Click on the *START ADVERTISING* button to start the procedure

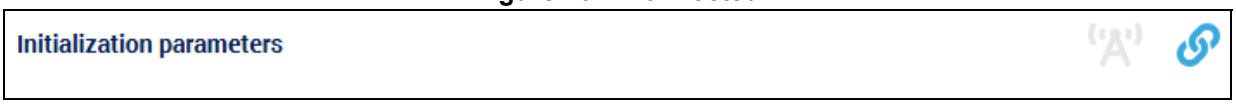
The search procedure starts, the advertising icon appears in blue, it is possible to stop it using the *STOP ADVERTISING* button.

Figure 101. Advertising



The connect icon may appear in blue if a remote device connects. In that case, advertising stops.

Figure 102. Connected



4 OpenThread mode

4.1 Presentation

4.1.1 Panel

The OpenThread main panel is organized with three tabs, *Commands*, *Scripts*, and *Network Explorer*.

Figure 103. OpenThread - Command tab

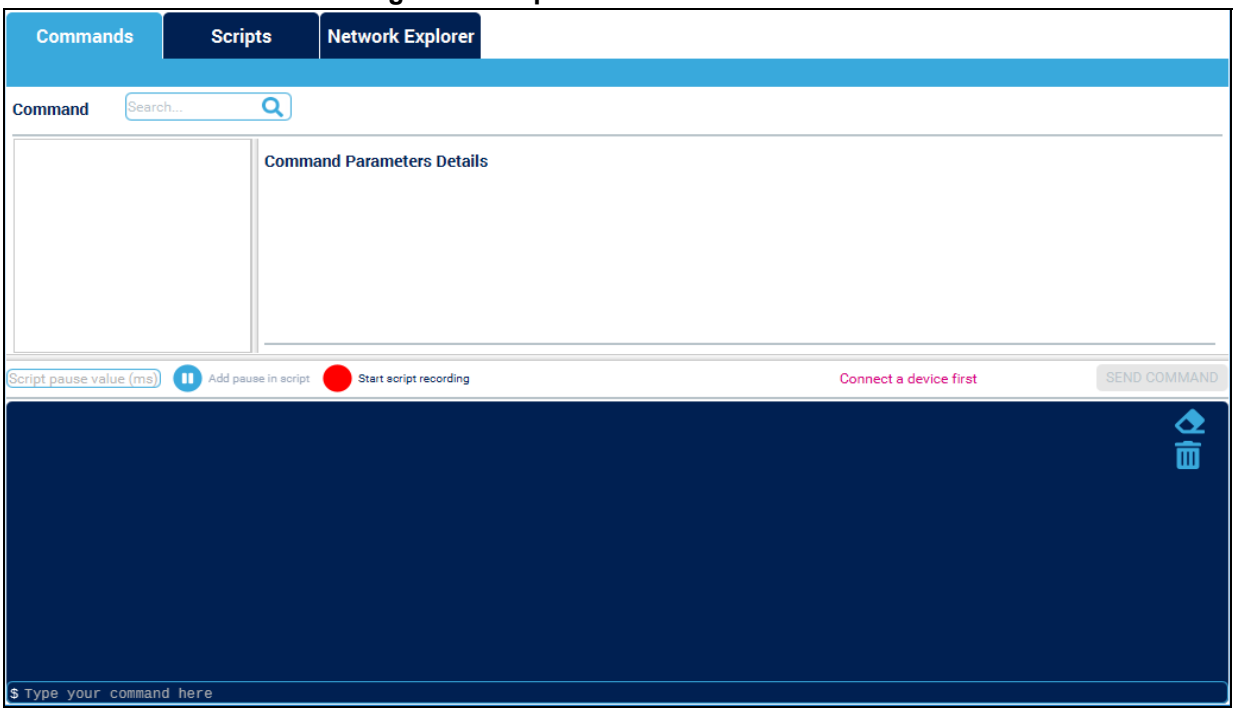
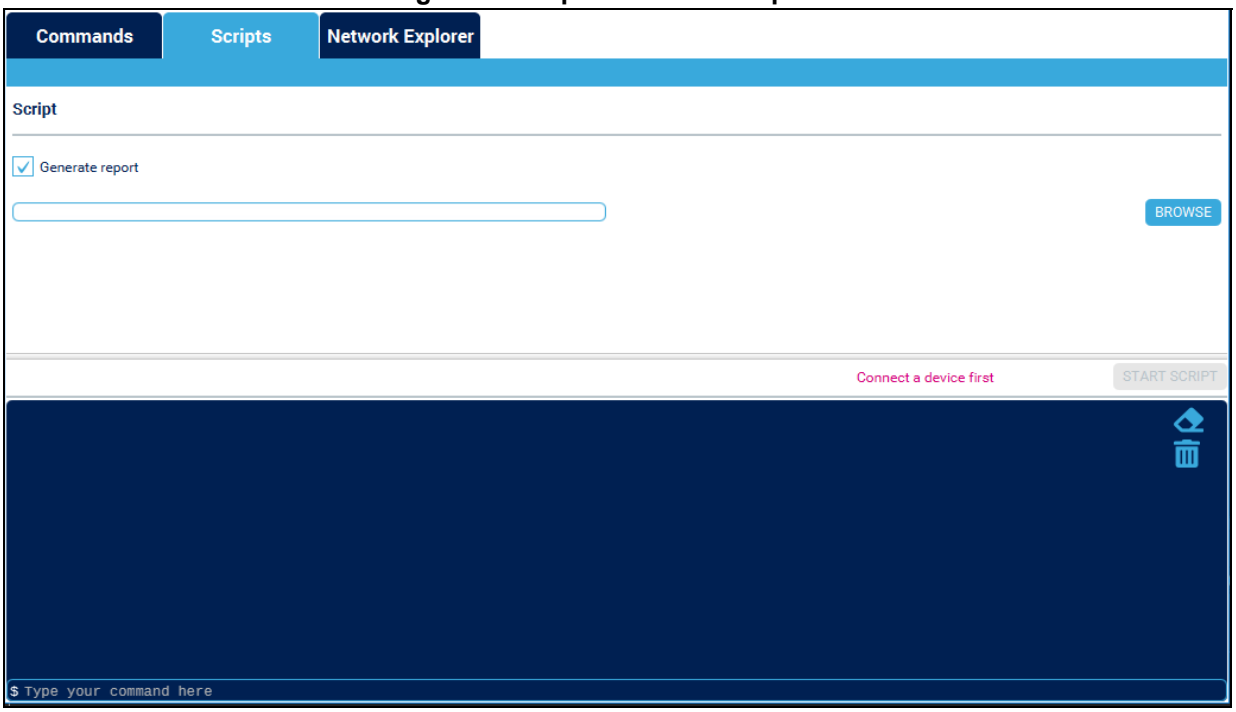


Figure 104. OpenThread - Script tab



The first two tabs have one common bottom area, the terminal area.

Figure 105. OpenThread - Network Explorer tab

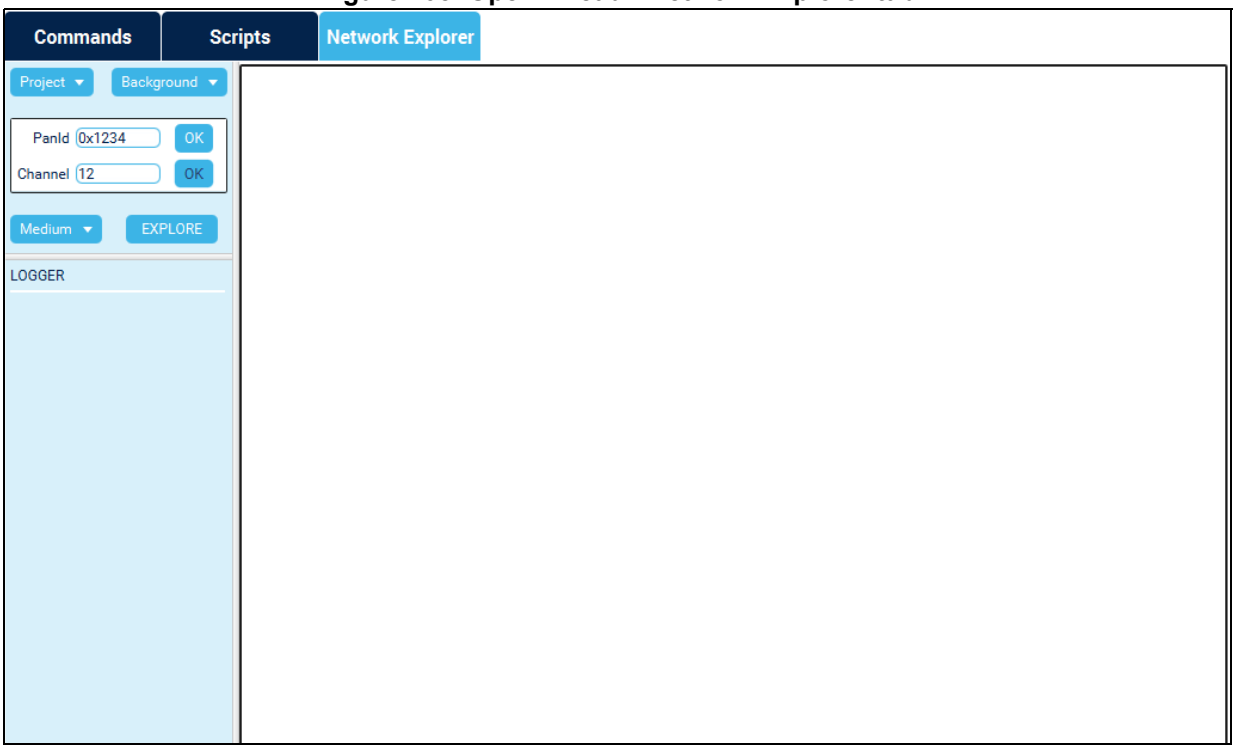


Figure 106. OpenThread common bottom area

The terminal area is used to show the messages exchanged between the application and the target. We can see the commands sent to the target and the responses received from the target. Those messages can be cleared with the rubber icon.

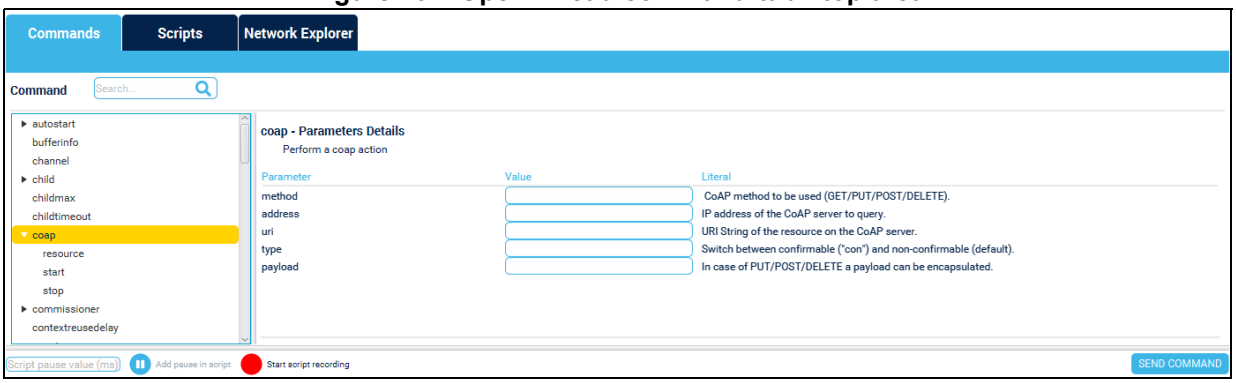
The bottom line with $ character is a command line. The user types the command with the parameters and presses <Enter> to send the command. The command sent with this line are recorded in the history file and can be recalled with up and down arrows. This history can be deleted with the trash icon.

One other way is using the commands list and parameter area to fill the line, then the user can modify the line and send a command with the entering key. The commands list and parameters area are described in the chapter Commands tab.

4.2 Commands tab

This tab is dedicated to the OT commands and parameters. The top area gives access to the commands list and parameters. Some commands can be used to read and send values, others are only commands sent to the OpenThread stack.

Figure 107. OpenThread command-tab - top area



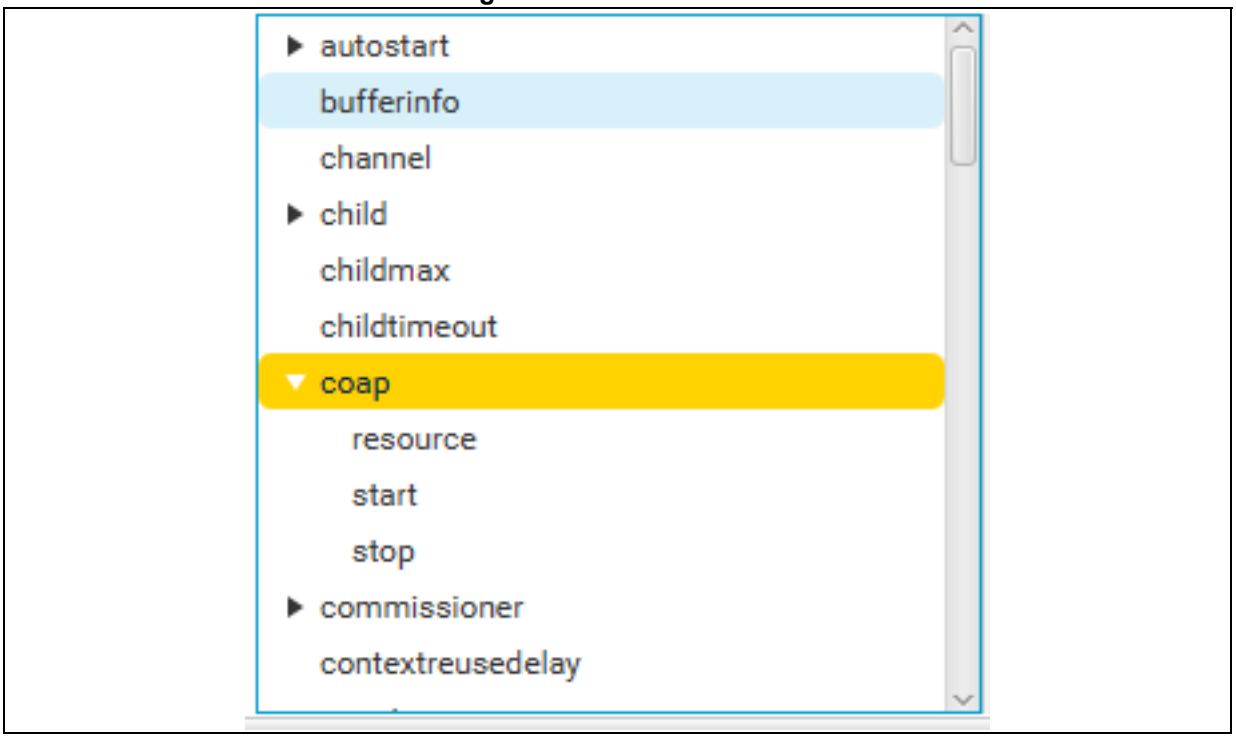
For commands used to send data, the *SEND COMMAND* button sends the command with parameters to the target.

For commands able to read information, two buttons are available: *READ*, *SEND*. The read button sends the command without parameters to read the value, the *SEND COMMAND* button sends the command with parameters to the target.

Figure 108. Script buttons

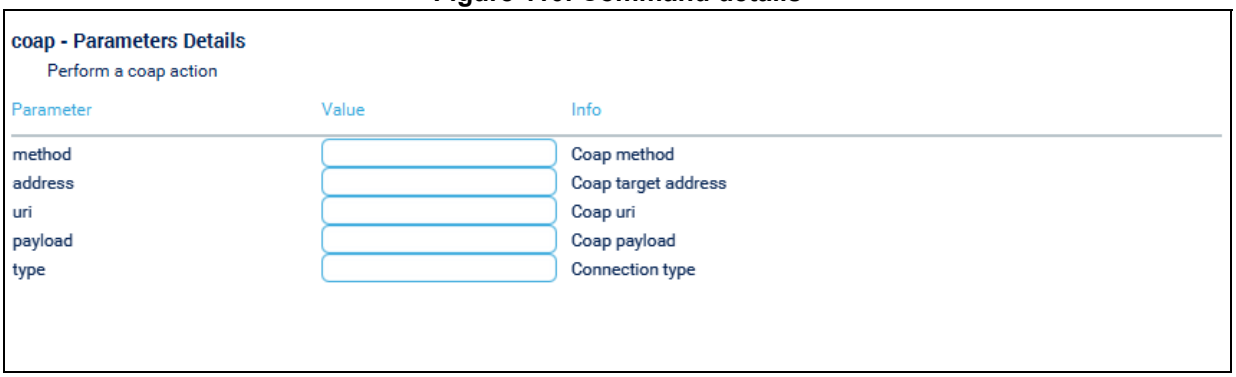
The *Start script recording* and *Add pause in script* buttons allow saving a script. This part is described in *Section 4.3: OpenThread scripts tab*.

Figure 109. Command list



The command list is arranged in alphabetical order, and accessible from the tree, for example below the *coap* command, there are *coap resource*, *coap start*, and *coap stop* commands.

Figure 110. Command details

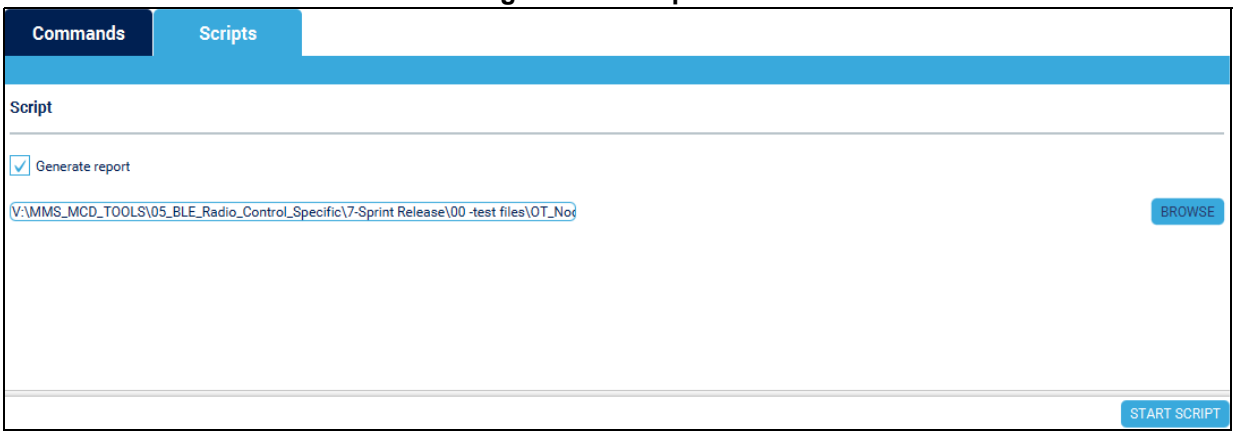


The command name and the definition are in the upper part of the command details area. Below is one table of parameters with the parameter name, there is one writable field to define the value and information concerning this parameter.

4.3 OpenThread scripts tab

The OpenThread scripts tab is used to launch the script stored in a text file.

Figure 111. Scripts tab



The scripts use the same syntax as Bluetooth low energy scripts. The OpenThread specificities are described in this chapter. Consult the Bluetooth low energy script description in *Section 3.4: Scripts on page 40* for general information.

4.3.1 OpenThread script example

Figure 112. Sample script

```
#STM32CubeMonitor-RF sample script
# OpenThread Ping Node Script

#Pause command
Pause ("Ready to start the test")

#Send reset command:
Send (reset)

#Set channel
Send (channel 11)

#Set the PAN ID:
Send (panid 0x1234)

#Bring up the IPv6 interface:
Send (ifconfig up)

#Start Thread protocol operation:
Send (thread start)

#Wait for a few seconds and verify that the device has become a
Thread Leader:
wait (5000)

#check state
Send (state)

#ipaddr
Send (ipaddr)
```

4.3.2 List of script commands

The OpenThread scripts use the same commands as Bluetooth low energy, but the Send command is modified to send Thread commands.

The OpenThread commands are sent with the instruction *Send*:

```
Send (OPENTHREAD_CMD_NAME Parameter1Value Parameter2Value)
```

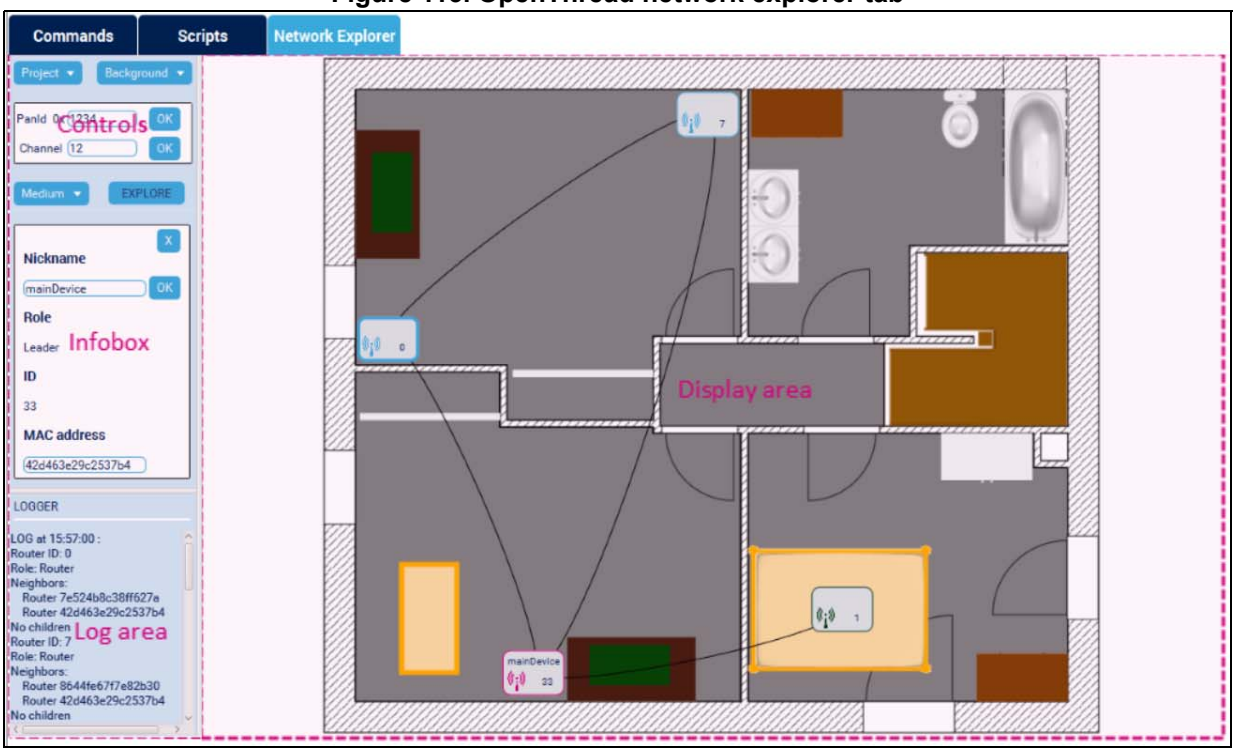
The part inside brackets is the command line to send.

4.4 Network Explorer tab

This feature can only be used if the DUT has the *Thread_Cli_cmd* firmware to be able to copy data from the UART to the OpenThread command-line interpreter. Refer to *Section 2.2.1* for further information about firmware.

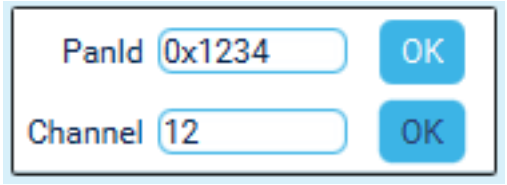
This tab is dedicated to the exploration and the display of the network to which the DUT device is attached. The representation of the network is displayed in the central area. There are some basic control functions in the up-left corner of the pane and just below there is information on the selected node in an Infobox plus logs of the exploration.

Figure 113. OpenThread network explorer tab



4.4.1 Controls

Figure 114. Auto connection functions



The network explorer tab easily configures the panId and channel of the DUT device.

The panId is entered in hexadecimal format, with no need to specify *0x*. The value must be contained between *0x0000* and *0xfffe*. The value *0xffff* means a non-configured panId.

The channel is defined in decimal format and must be contained in [11;26]. The panId must be configured before configuring the channel.

For both parameters, if the filled value is in the wrong format, nothing is changed, and the actual value of the device remains displayed. Moreover, if a network exploration is ongoing neither parameter can be changed.

At the first connection of the DUT or when switching to the network explorer tab, the tool checks the current values of both parameters and displays them in the fields as information.

Figure 115. Project and background management

The two menu buttons on the top of the control area give control to the project itself and the background image.

The Project menu proposes three choices:

1. The New choice cleans the current session by resetting the display area and stopping the ongoing exploration if there is one.
2. The Open choice opens a file explorer to choose a backup of a project to use in the session. When a project is loaded from the Open choice box there is a two-step process:
   – The saved image is first restored as the background of the right area.
   – Then when a scan is started, if a device that has been saved in the project is detected, it is instantaneously displayed at its last place with its former nickname. This association is based on the unique MAC addresses of the devices.
3. Save choices save the current project. In this backup, there are the background image, the location of the icons on that image, and the nicknames of the devices.

The Background menu allows either to:

- Remove the background image.
- Open a file explorer to put an image in the display area as a background.

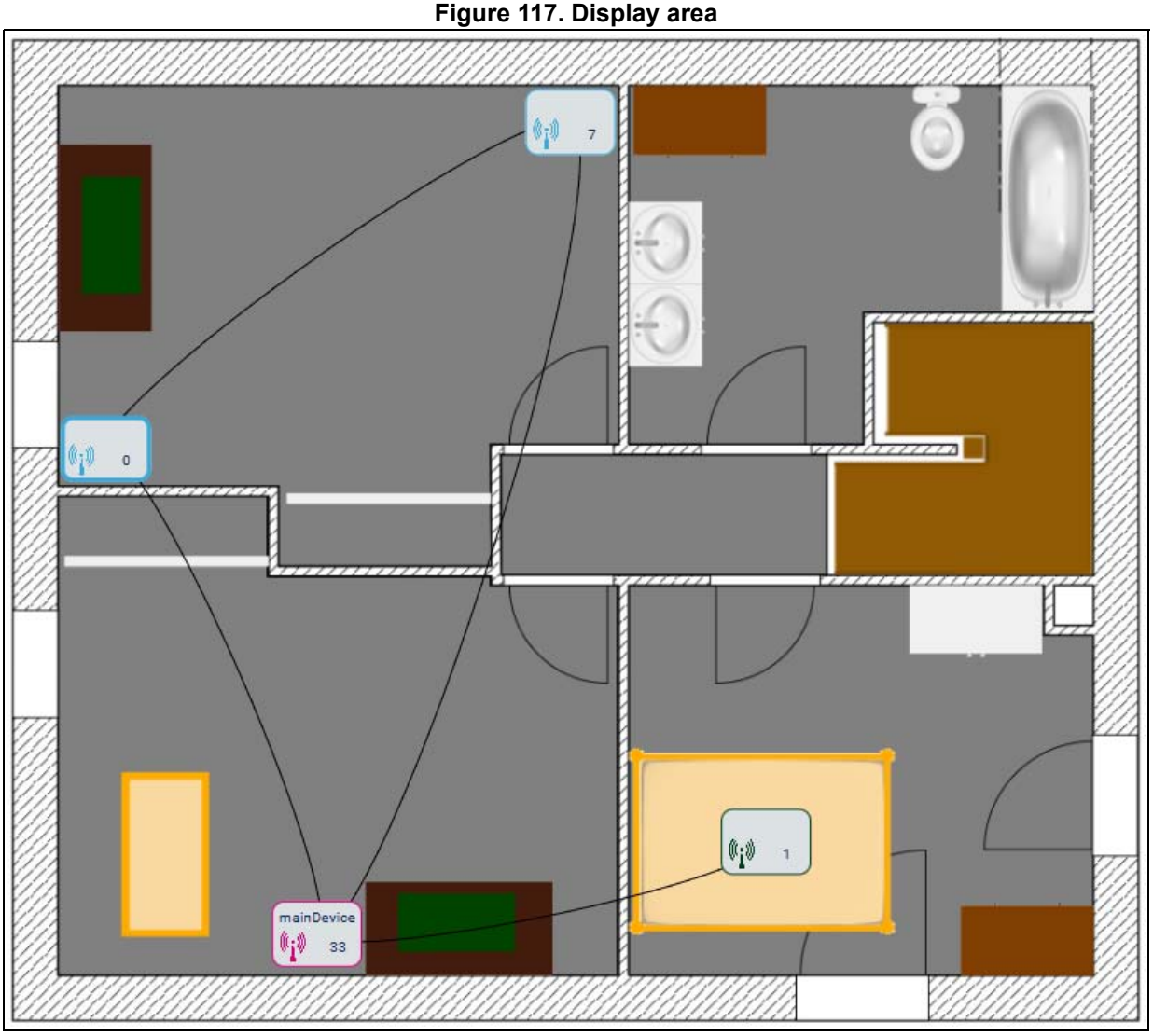
Figure 116. Explore and size choice controls

Once the DUT device is connected to a Thread network, the EXPLORE button starts the network exploration sequences. It turns to *STOP* when the exploration is ongoing.

The choice box at the left of the EXPLORE button allows choosing the size of the icon between three standard sizes: *Small*, *Medium*, and *Large*. It can be changed at any time. The size of the icons is adapted according to the dimensions of the background image.

4.4.2 Display area

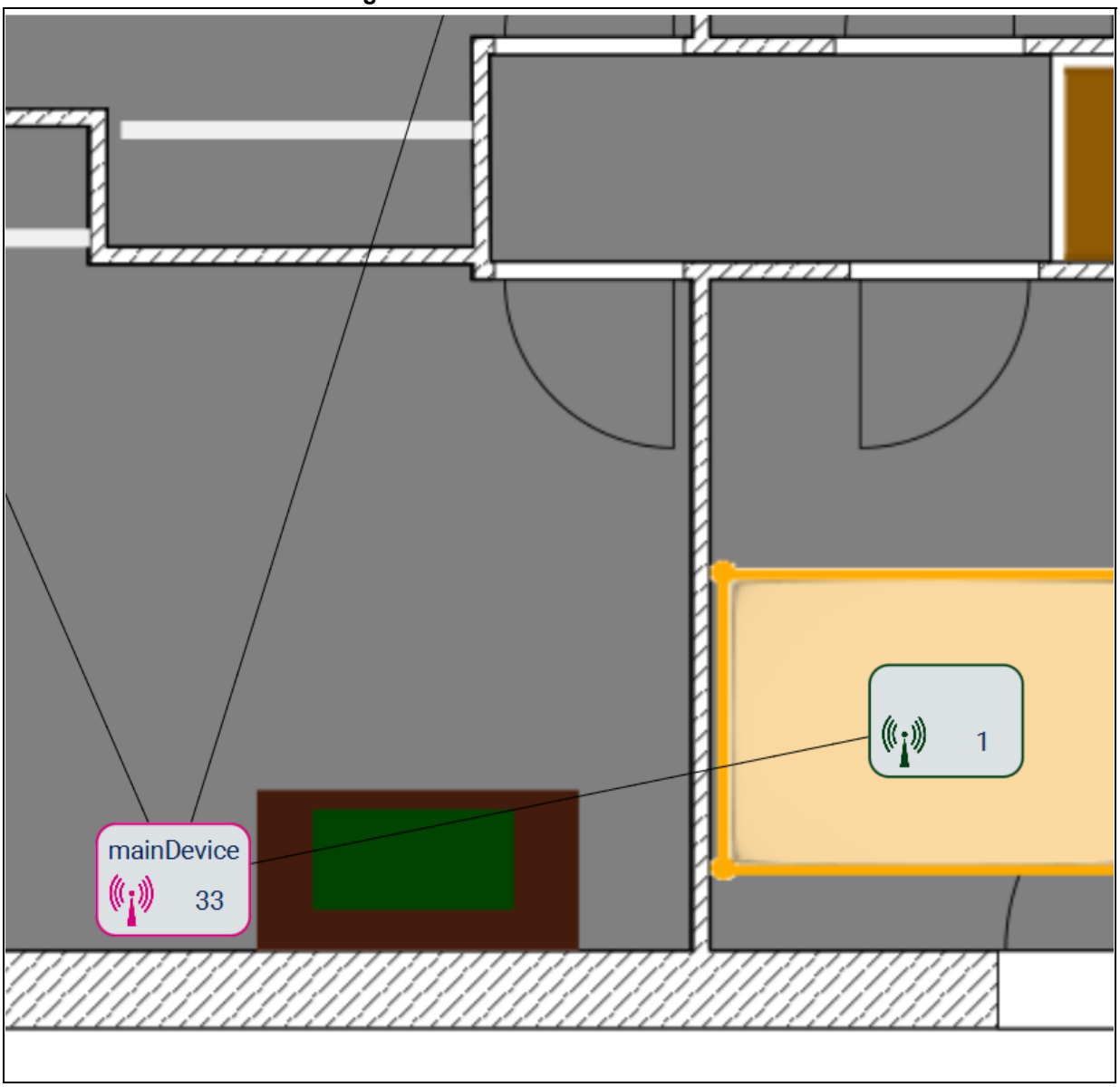
Figure 117. Display area



The result of the network exploration is displayed with icons representing the devices of the network and their links. Each icon gives 3 types of information:

1. The role is given by the color of the borders and the logo (pink for a Leader, cyan for a Router, and green for a Child).
2. The ID is given by the number on the right side of the logo.
3. The eventual nickname is written above the logo.

Figure 118. Zoom and motion controls

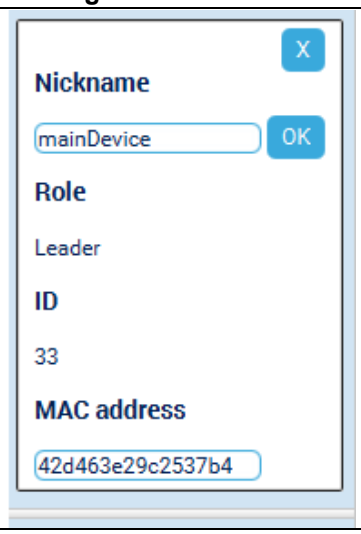


In the display area, it is possible to make several kinds of movement:

- An icon can be moved everywhere inside the right area by simply holding the left click of the mouse. It turns gray if dragged on another icon and is automatically replaced if dropped on another icon to avoid overlays.
- Zoom in or out is done with the mouse wheel. The motion is centered on the mouse pointer.
- The whole content of the right area can be moved by holding the right click of the mouse. There are constraints to this movement though because what defines the background of the area (imported image or default blank background) cannot go completely off the area.
- A double click (left) anywhere on the area centers the background and restores zoom x1.

4.4.3 Infobox

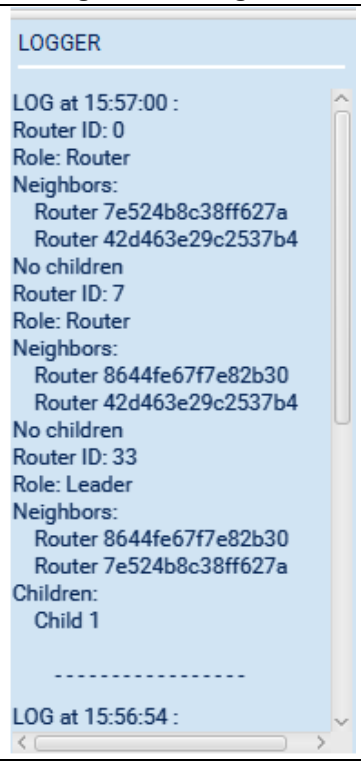
Figure 119. Infobox



An Infobox can be instantiated just below the control area by clicking on the concerned icon. It allows modification of the node nickname and indicates its role, ID, and MAC address.

4.4.4 Log area

Figure 120. Log area



The log area is in the bottom-left part of the tab. It prints the last two exploration results of the network in a written form. This area is updated after each new exploration.

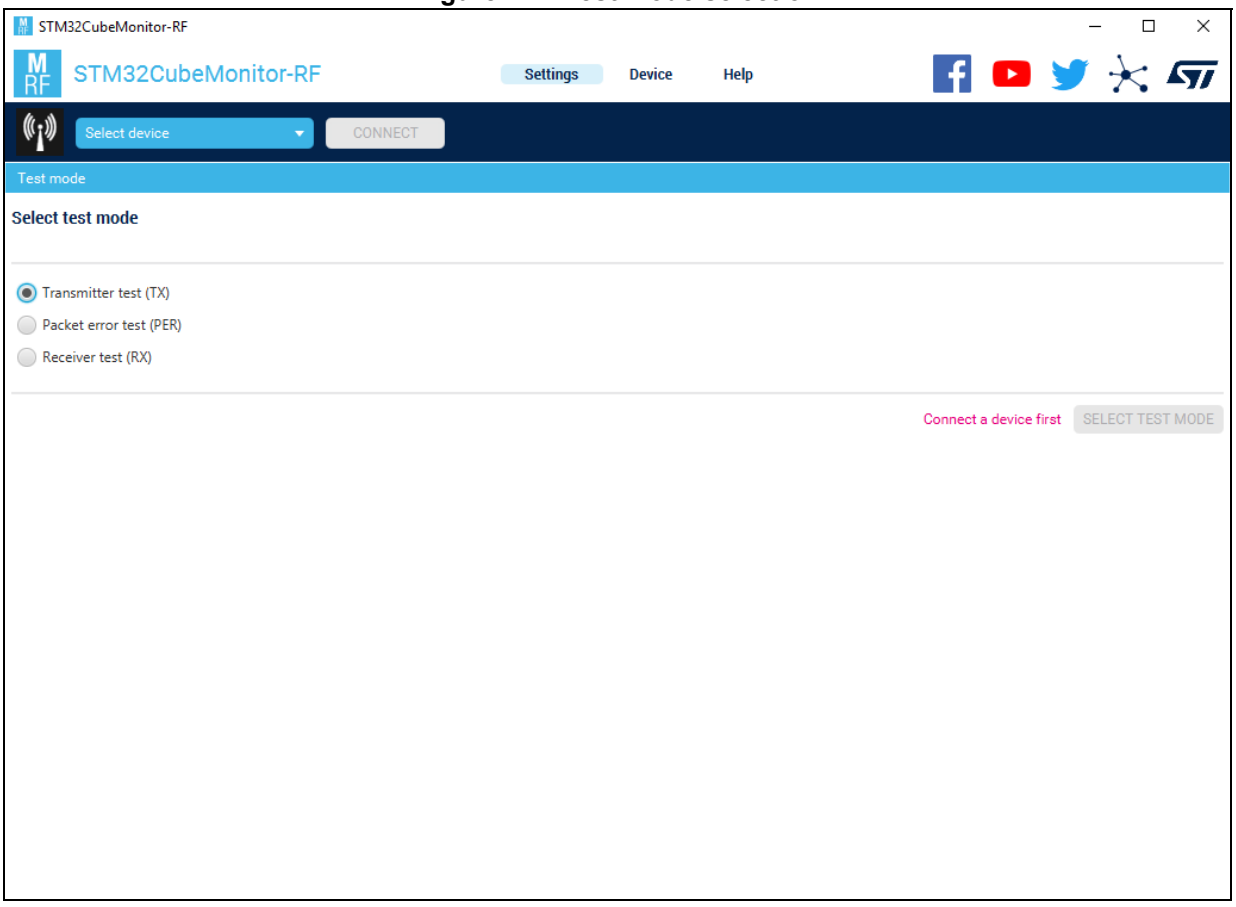
5 802.15.4 RF test mode

5.1 Presentation

The RF test panel performs the 802.15.4 radio-frequency tests on the main device. Three test modes are available: transmitter (TX), receiver (RX), and packet error rate (PER):

- The TX test sets the device in emission (TX continuous).
- The RX test sets the device in reception.
- The PER test sets the device in reception and one additional device is used as a packet generator.

Figure 121. Test mode selection



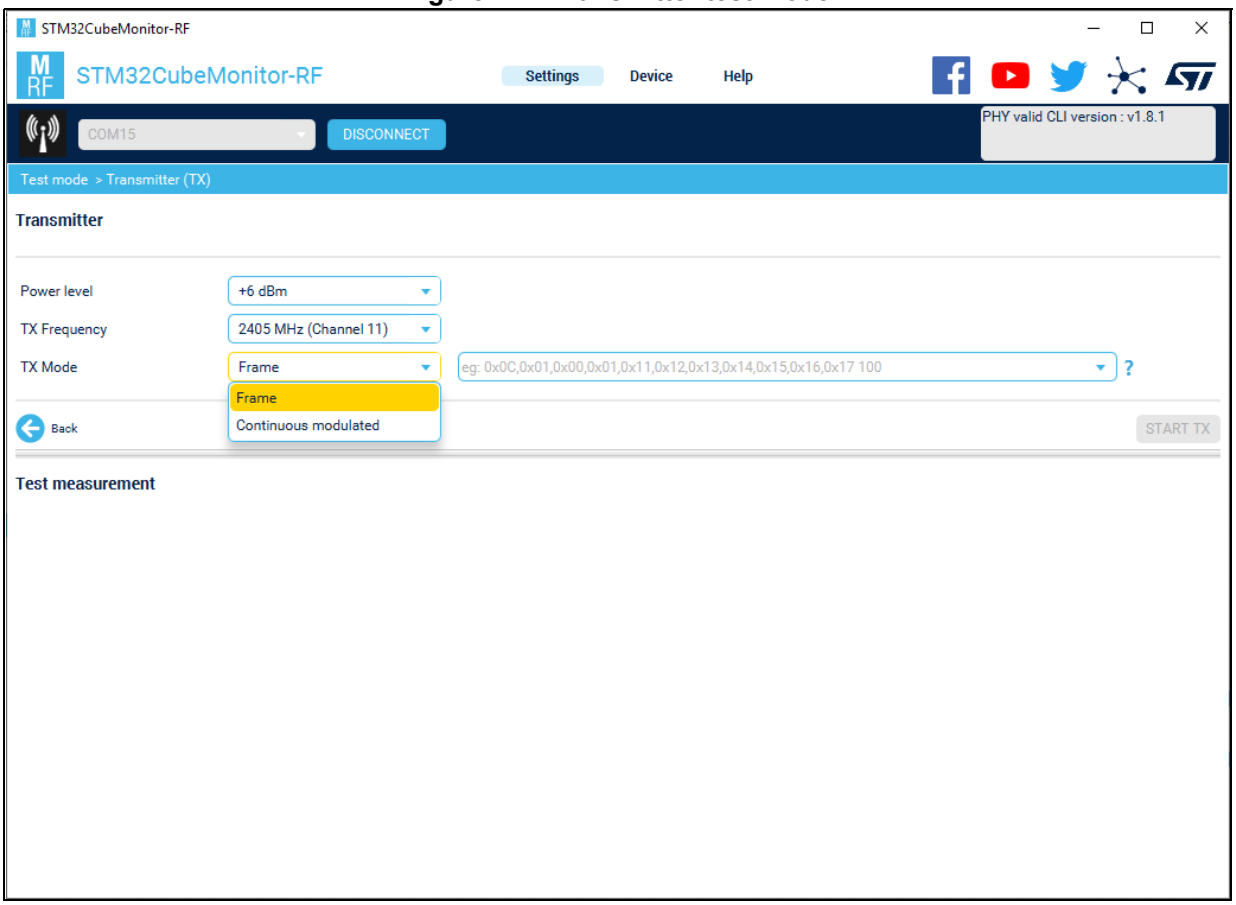
The user selects the mode by checking the radio button and press the *SELECT TEST MODE* key to switch on the new panel.

To change the mode, it is necessary to come back to this panel. There are a *Back* key and a breadcrumb link in each test panel to come back to this *Test mode selection* panel.

5.2 Transmitter test (TX) mode

This test mode configures the 802.15.4 device in emission. Two TX modes are available, Frame mode and Continuous modulated mode.

Figure 122. Transmitter test mode



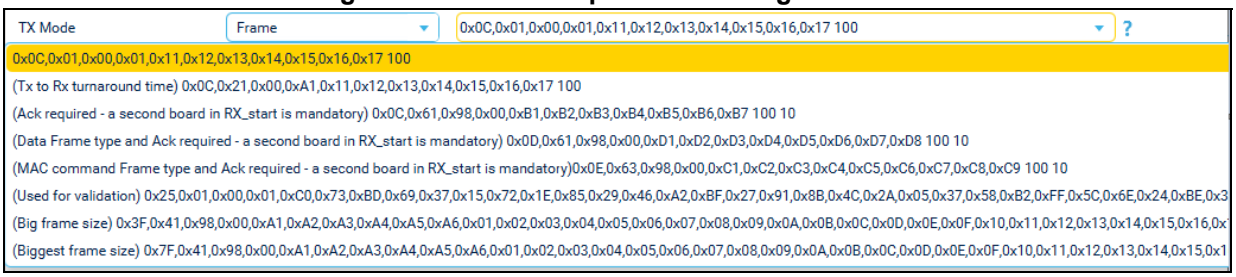
The user must:

- Select the power level (+6 dBm to -21 dBm).
- Select the TX frequency (channel11 - 2405 MHz to channel 26 - 2480 MHz).
- Select the TX mode, Frame, or Continuous modulated.

5.2.1 Frame mode

This mode allows the user to send a MAC frame. Either the user selects one frame available in the pick list or it fills itself the field.

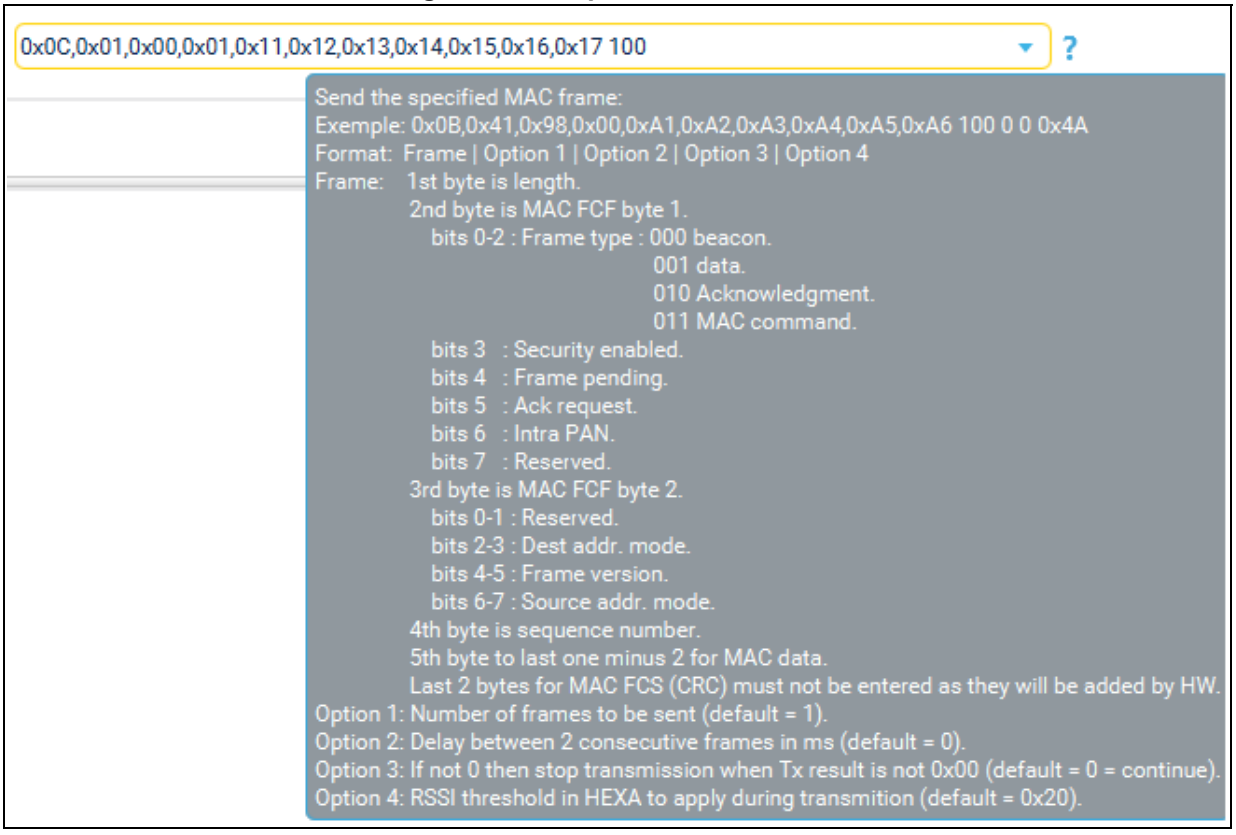
Figure 123. Field and picklist defining the frame



Note: In the picklist, there is the frame required for Tx to Rx turnaround time certification test.

The help information is visible with a mouse-over in the question mark on the right side of the field.

Figure 124. Help frame information



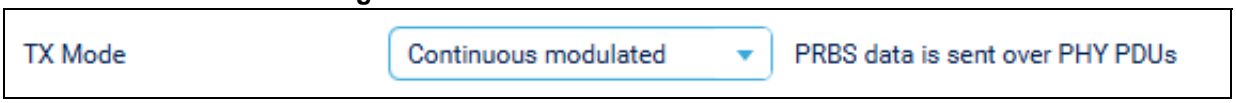
The *START TX* button is enabled when the frame in the field is valid. Press the start button to launch the transmission, the button is disabled until the frame is transmitted.

5.2.2 Continuous modulated mode

This mode transmits a continuous signal where pseudo-random binary sequence (PRBS) data is sent over PHY PDU.

Press the *START TX* button to launch the transmission. The label of the button is switched to *STOP TX* and allows the user to stop the transmission.

Figure 125. Continuous modulated test mode



5.3 Receiver test (RX) mode

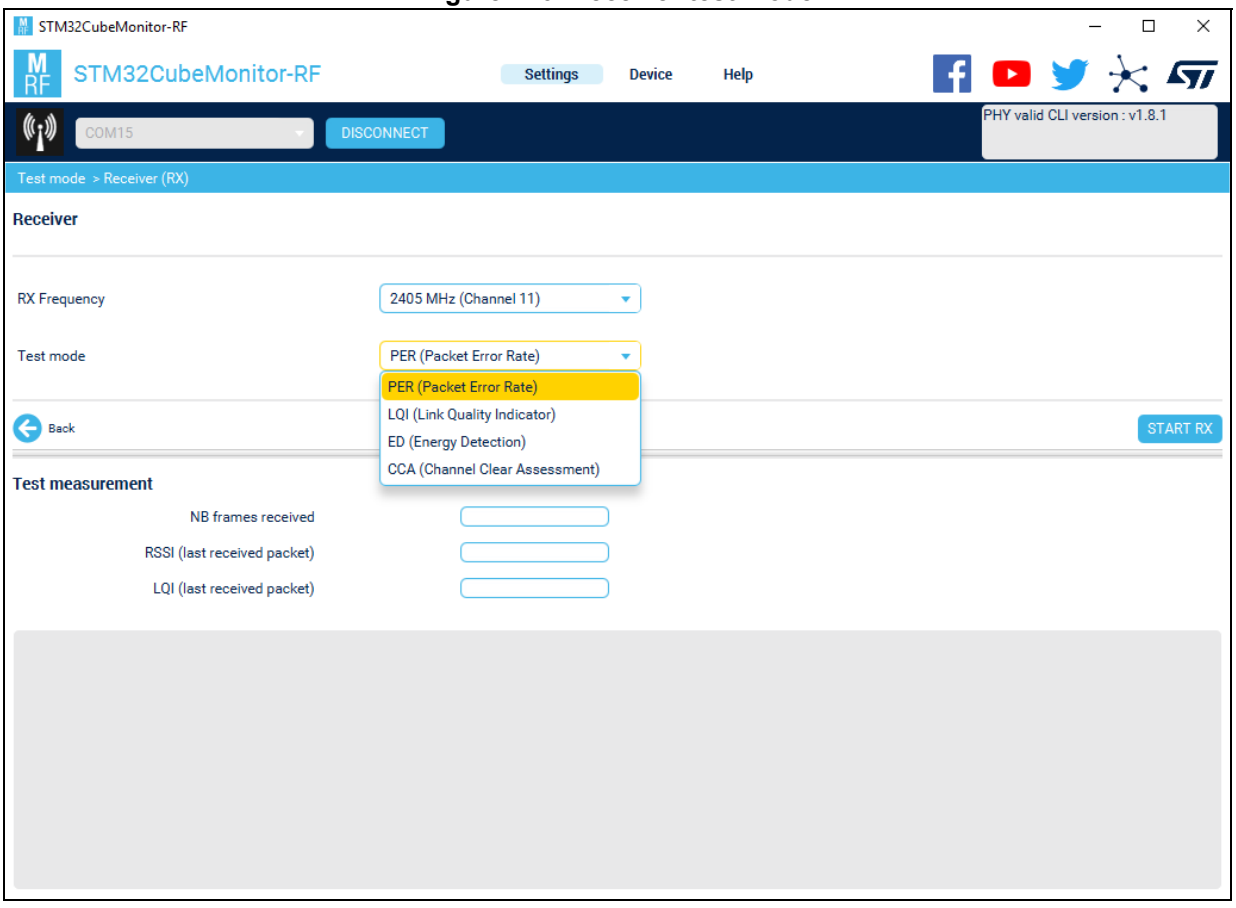
This test mode configures the device in reception and requires an external generator.

Four tests are available:

1. PER (Packet Error Rate): Requires an external frame generator.
2. LQI (Link Quality Indicator): Requires an external frame generator.
3. ED (Energy Detection): Requires an external continuous wave generator.
4. CCA (Channel Clear Assessment): Requires an external frame generator.

Note: LQI, ED, and CCA tests are available with PHY valid CLI version v1.8.1 and upper.

Figure 126. Receiver test mode

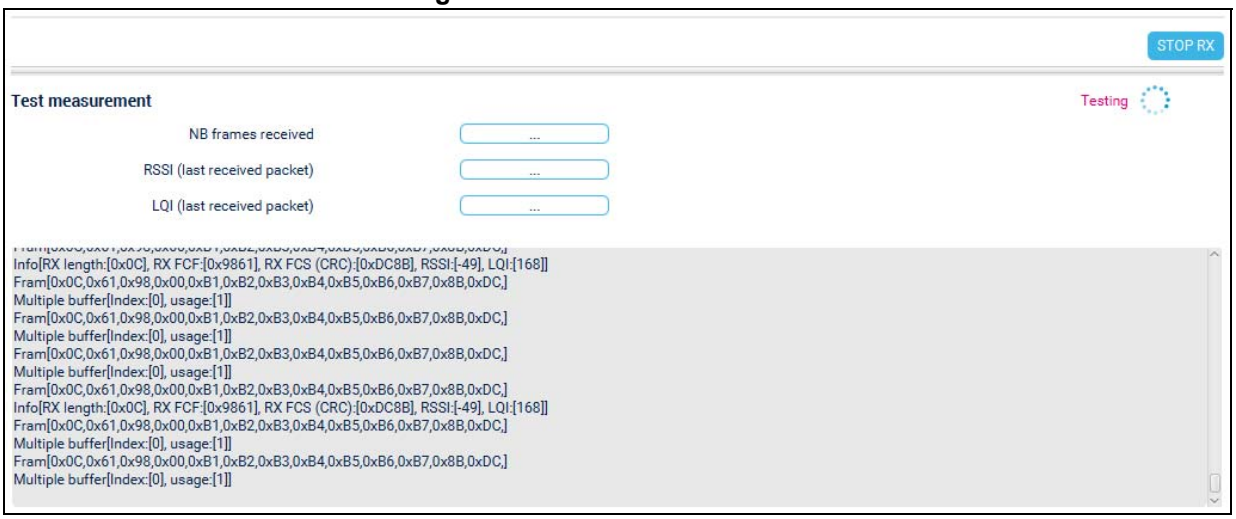


5.3.1 Packet error rate (PER) test

This test requires an external frame generator and to follow the procedure as below:

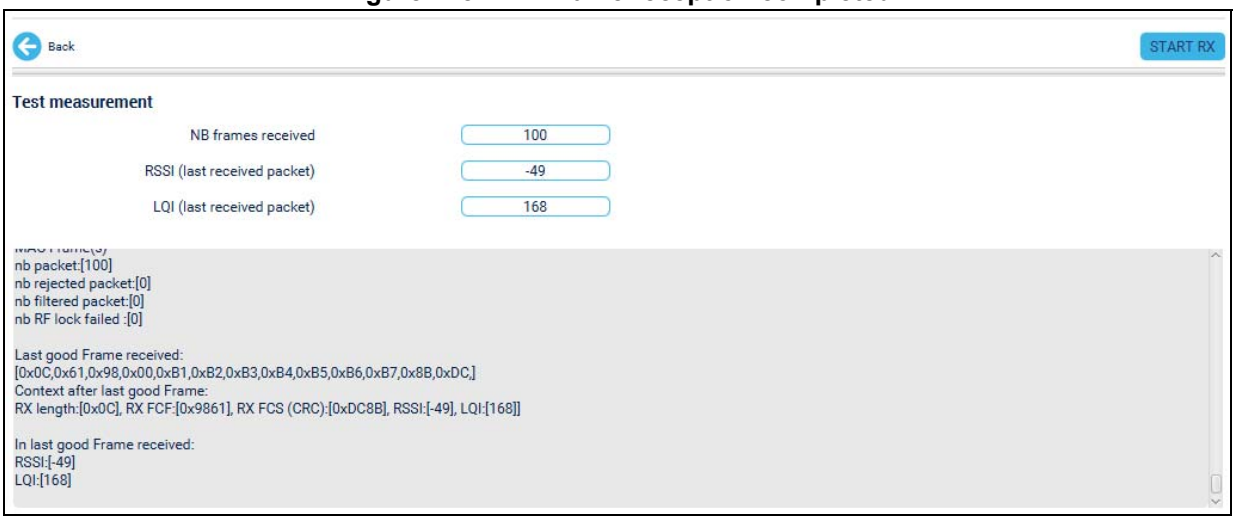
- Select the channel to be tested.
- Press the *START RX* button; the device enters in receiver mode and the button switch to *STOP RX*.
- With one external generator, send the frames to test in the frequency selected above.
- On the application side, the frames received appear in a gray part. This part is available from PHY valid CLI version v1.8.0 and upper.

Figure 127. PER frames received



- Once the frames are completely sent, press the *STOP RX* button. The three fields NB frames, RSSI, and LQI are filed. The button switches to *START RX*.

Figure 128. PER frame reception completed



- According to the number of frames sent, the PER can be calculated with the value in the *NB frames received* field.

5.3.2 Link quality assessment (LQI) test

This test requires an external frame generator and to follow the procedure as below:

- Select the channel to be tested.
- Either the measurement is done in continuous (default mode) or step by step by checking the *Single measurement* item.
- With one external generator, send the RF signal to test in the frequency selected above.
- Press the *START RX* button to launch the LQI measurement.
- The instantaneous measurement appears on the right side and is also reported in the chart.

Figure 129. LQI measurement

5.3.3 Energy detection (ED) test

This test requires an external frame generator and to follow the procedure as below:

- Select the channel to be tested.
- Either the measurement is done in continuous (default mode) or step by step by checking the *Single measurement* item.
- With one external generator, send the RF CW signal in the frequency selected above.
- Press the *START RX* button to launch the ED measurement.

The instantaneous measurement appears on the right side and is also reported in the chart.

Figure 130. ED measurement

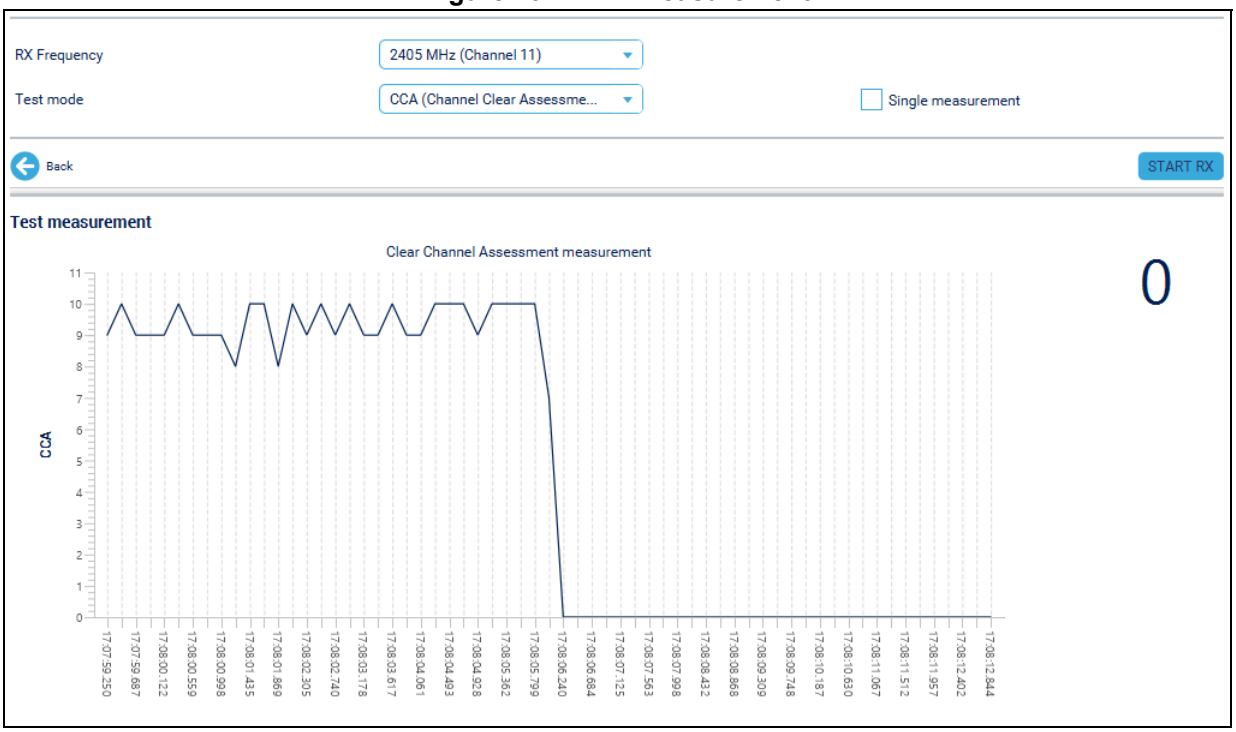
5.3.4 Channel clear assessment (CCA) test

This test requires an external frame generator and to follow the procedure as below:

- Select the channel to be tested.
- Either the measurement is done in continuous (default mode) or step by step by checking the *Single measurement* item.
- With one external generator, send the RF signal to test in the frequency selected above.
- Press the *START RX* button to launch the CCA measurement.

The instantaneous measurement appears on the right side and is also reported in the chart.

Figure 131. CCA measurement



5.4 Packet error rate (PER) mode

This mode configures the device in reception and one other device to play the role of the generator.

The tool makes three measurements:

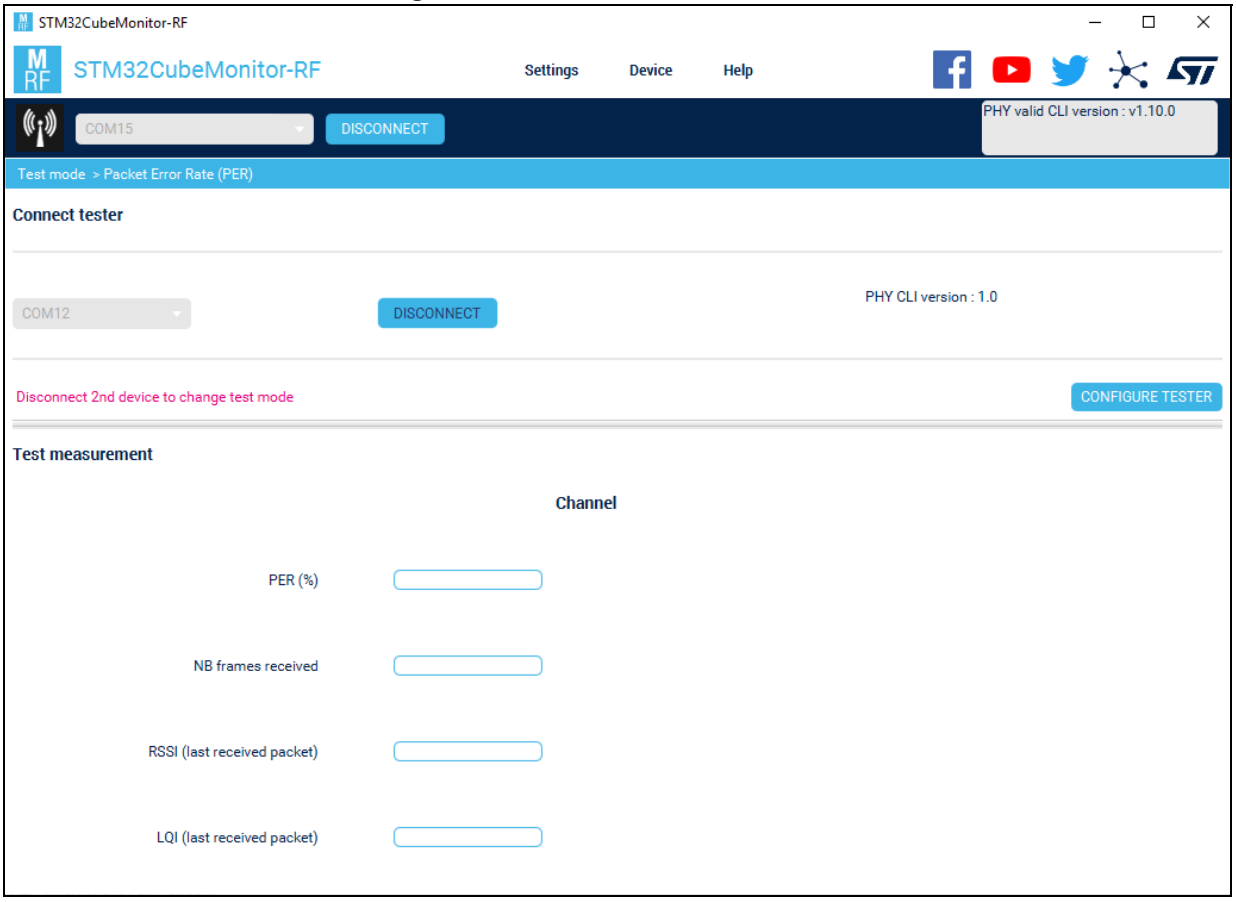
- RSSI: Received Signal Strength Indication
- LQI: Link Quality Indicator
- PER: Packet Error Rate - computed with the number of frames received and the number of frames sent
  100 x (Number of Frames sent - number of frames received) / Number of Frames sent

Four steps are necessary:

- Connect the additional device for playing the role of a packet generator (tester).
- Configure the parameters of the tester.
- Configure the parameters of the device under test (DUT).
- Configure the measurement.

5.4.1 Connecting the additional device to play the role of a packet generator (tester).

Figure 132. Packet tester connection



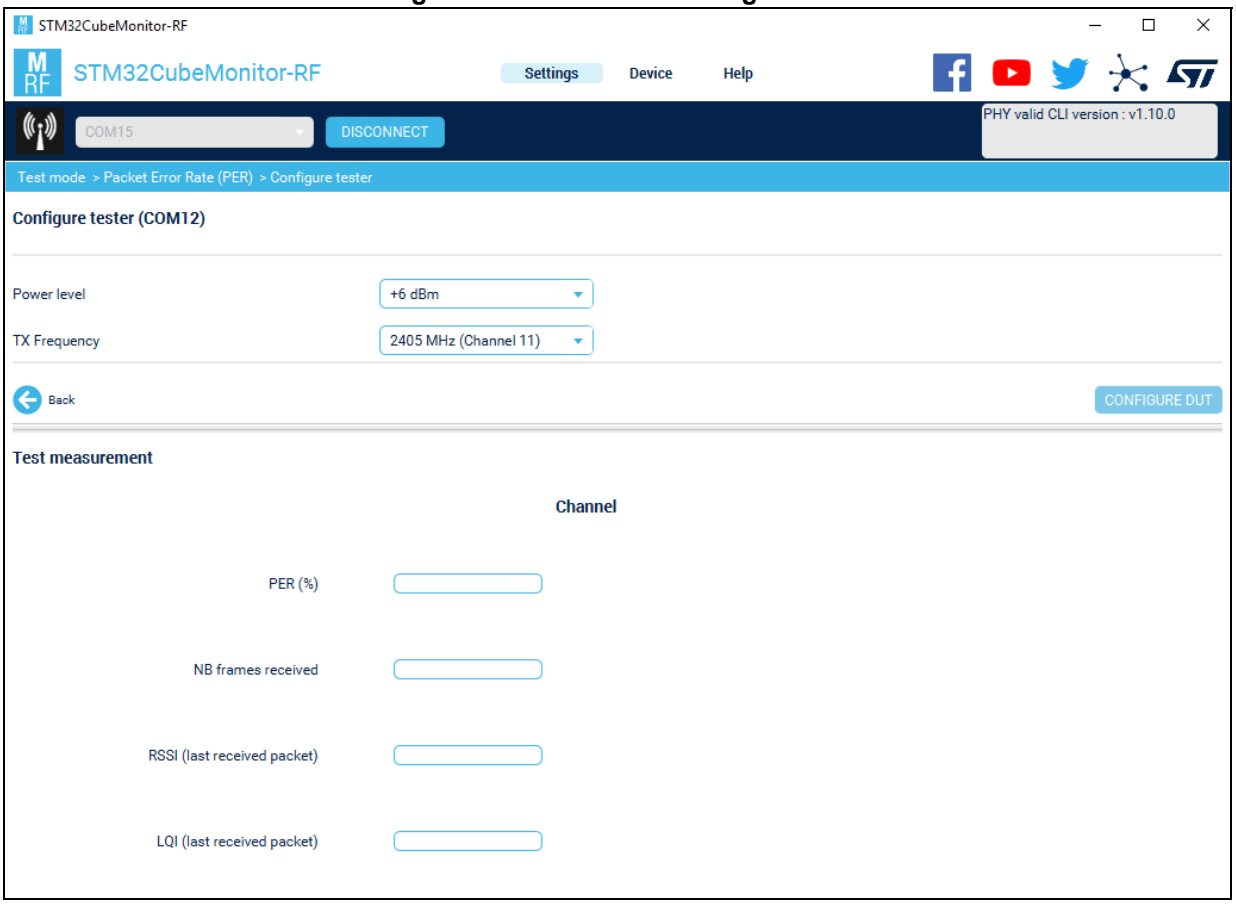
- Plug one additional device in the computer (same requirements as the first device, refer to *Section 2.2*).
- Select the serial port to use in the picklist.
- Click on the *CONNECT* key, the device information must appear on the right side of the *connect* key.

When the second device is connected, it is not possible to change mode. The user needs to disconnect the device first and then use the *back* button.

Click on *CONFIGURE TESTER* to set the tester parameters.

5.4.2 Configure the parameters of the tester.

Figure 133. PER tester configuration



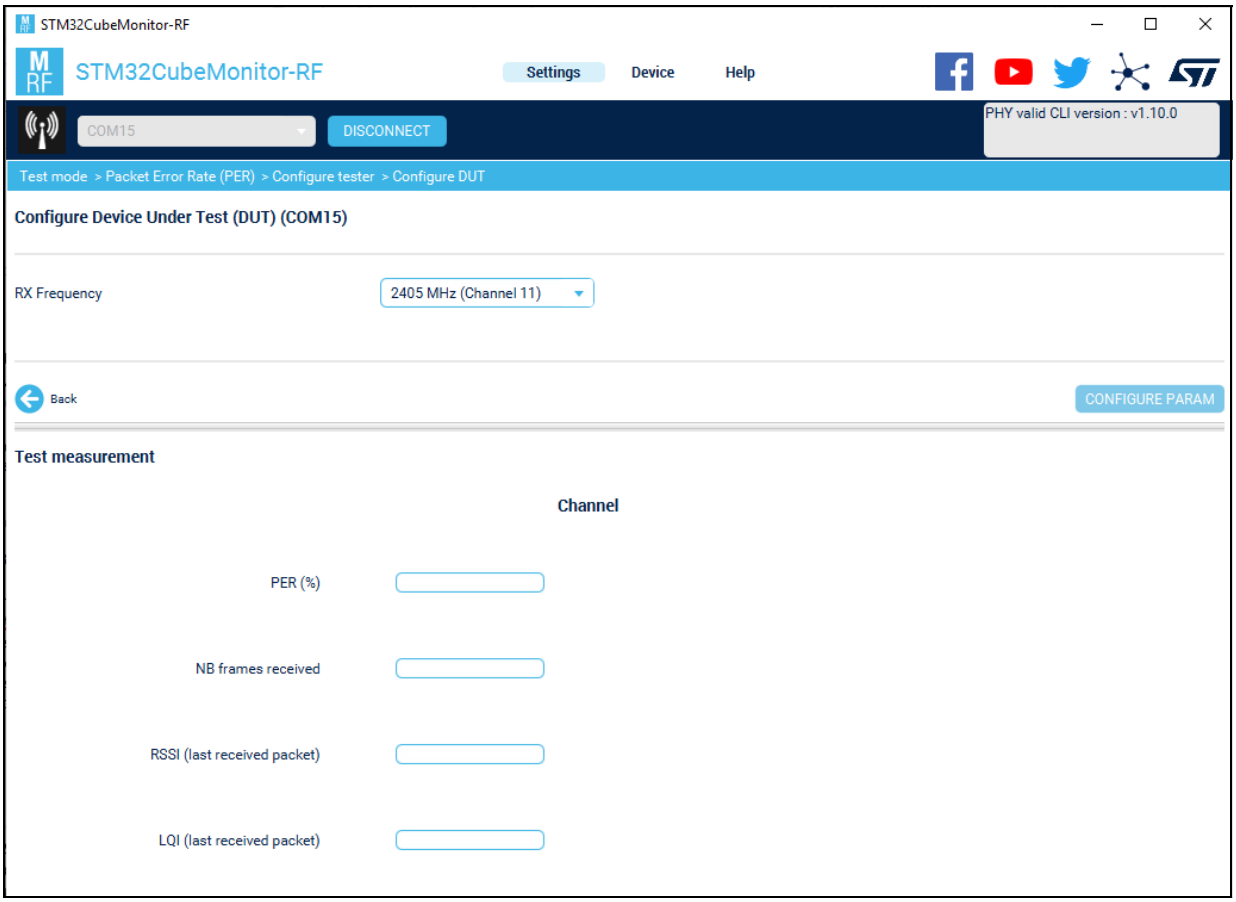
The user must:

- Select the power level in the pick-list *Power Level*.
- Select the frequency in the pick-list *TX frequency*. This parameter is used only for the single measurement mode, it is not used for continuous or multiple channel modes. It is applied to the tester device.

Click on *CONFIGURE DUT* to set the Device Under Test configuration.

5.4.3 Configure the parameters of the device under test (DUT).

Figure 134. DUT configuration



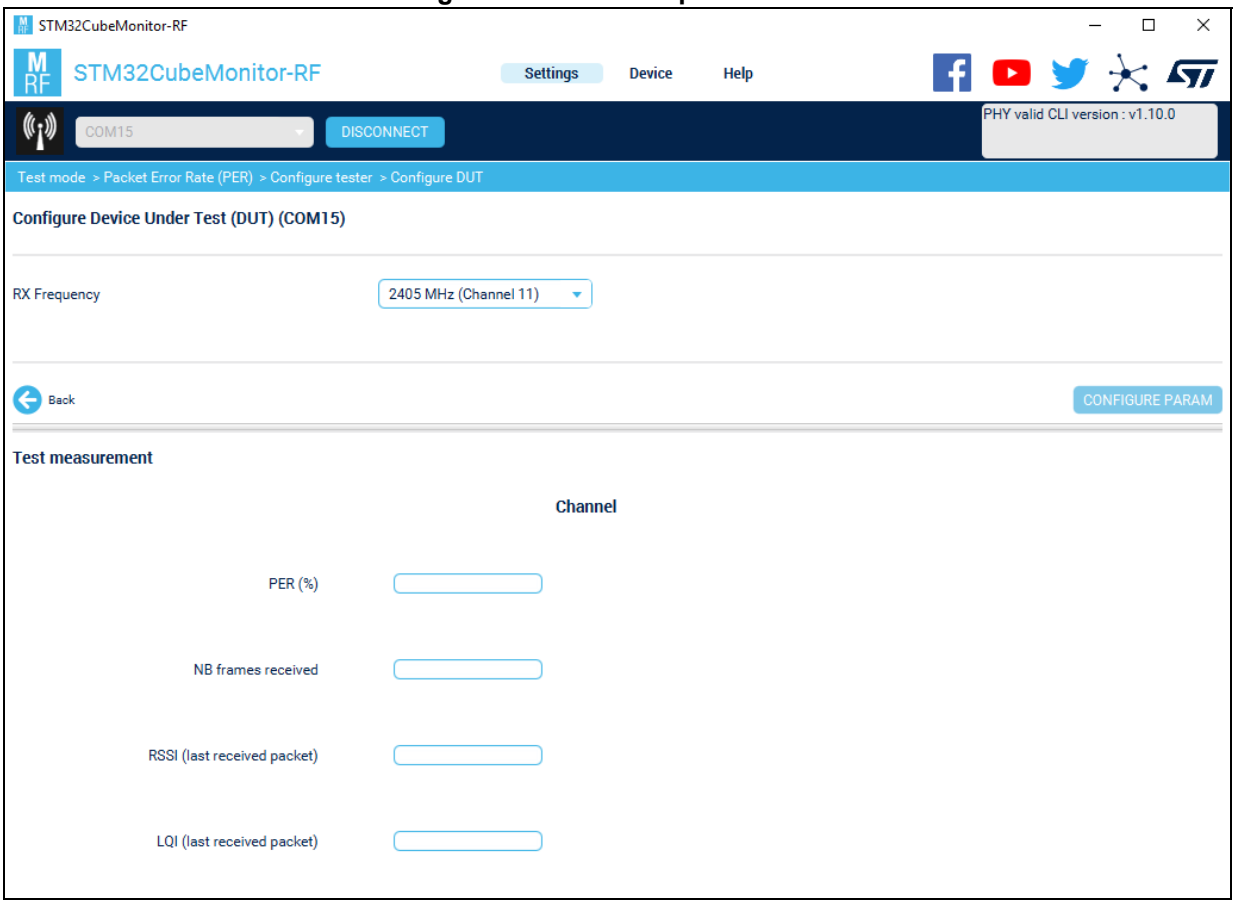
The user must:

- Select the frequency in the pick-list *RX frequency*. It is the frequency of the DUT.

Click on *CONFIGURE PARAM* to set the test configuration:

5.4.4 Configure the measurement.

Figure 135. PER test parameters



Three measurement modes are available:

- **Single measurement** measures once the frame number is defined. The frequency of the tester is the one defined in panel *PER tester configuration* (TX frequency). The frequencies of DUT are as defined in the *DUT configuration* panel.
- **Continuous measurement** repeats the measurement on frames number until the user presses the *Stop Test* key. The frequency of DUT and Tester is the same, it is the one defined in the panel DUT configuration.
- **Multiple channels** measure the frequency defined in the *Fill channel list* field. The default values are *11-26* this means all channels in the range 11 to 26. It is possible to use a comma to define channel by channel: *12,15,24* or mix both: *11,14-20,25,26*. The user can interrupt the test with the *Stop Test* key.

The result of continuous and multiple channel measurements can be saved in a csv file. The user must check the *Save test verdict in file* checkbox and must define the name of the file by the *SELECT FILE* key before starting the test.

Table 3. Measurement setting

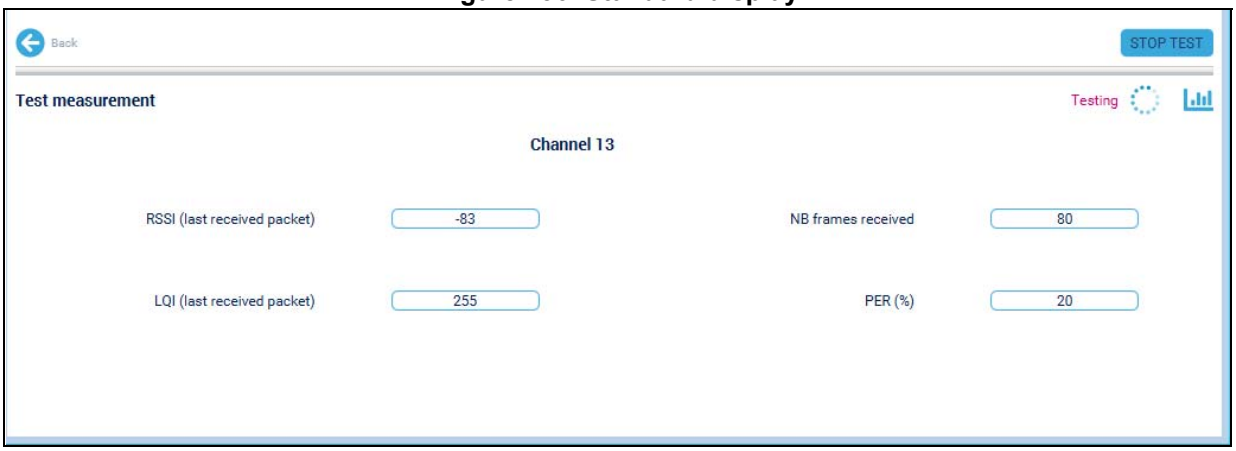
Measurements	Single	Continuous	Multiple channels
Continuous measurement checkbox	Unchecked	Checked	Checked
Multiple Channels checkbox	Unchecked	Unchecked	Checked
Save test verdict in file checkbox	Not available	Available	Available

Three display modes are available:

1. Standard display

There are the PER and RSSI values, and LQI for one channel.

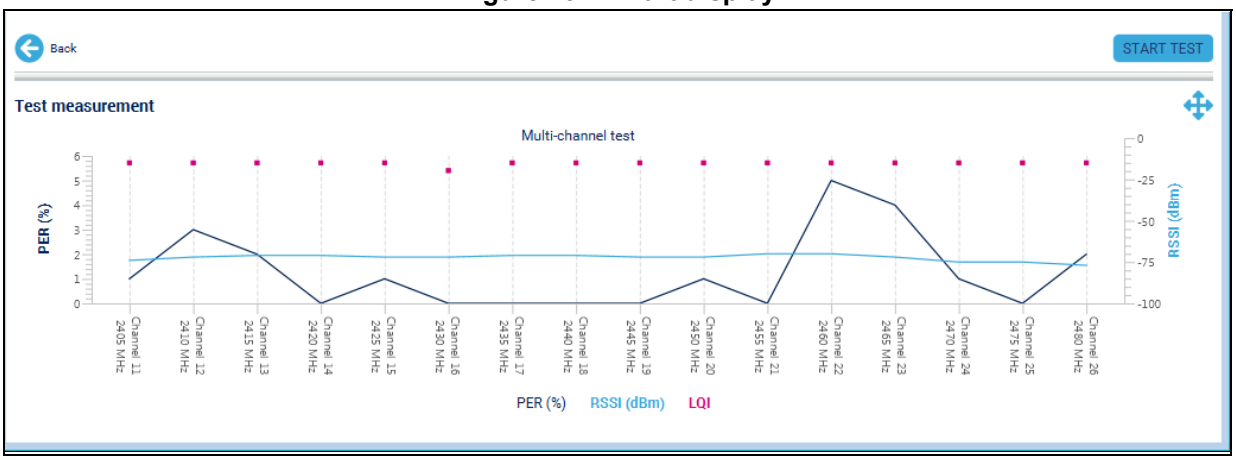
Figure 136. Standard display



2. Chart display

In the same chart, there are the PER value, RSSI value, and LQI for channels that the user defines.

Figure 137. Chart display



3. Large display

It is possible to switch from a PER display to RSSI and LQI with the arrow icons on the left or right side.

Figure 138. Large PER display

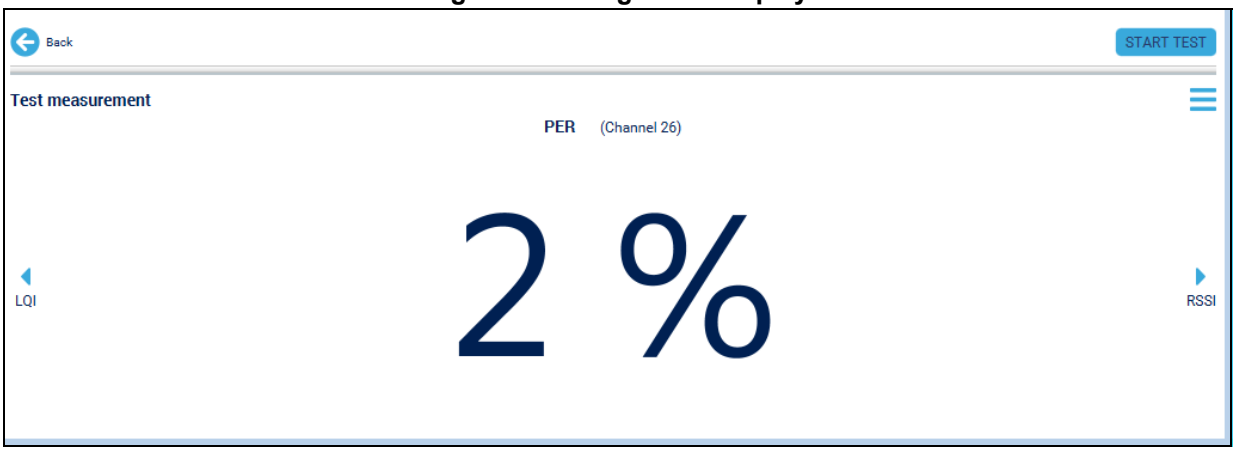
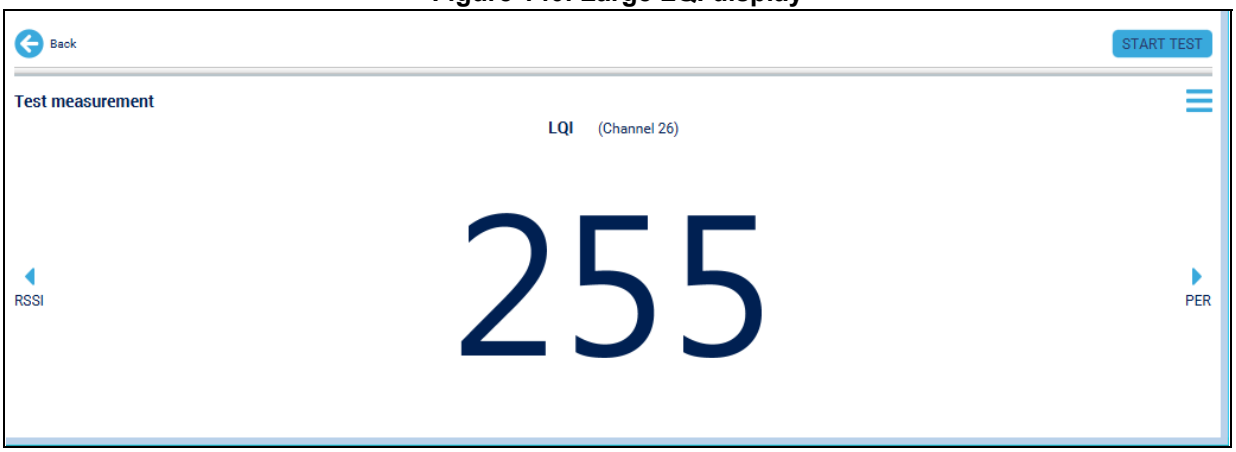


Figure 139. Large RSSI display

Figure 140. Large LQI display



6 802.15.4 Sniffer

6.1 Presentation

The 802.15.4 sniffer allows the user to detect and log 802.15.4 packets between the devices communicating in the neighborhood of the sniffer device. Packets captured by the device are logged and formatted in a readable format, thanks to Wireshark, an external free software tool.

6.2 Prerequisite

6.2.1 Sniffer device

To configure the device as a sniffer, refer to *Section 2.2.3: VCP device*. Once done, connect the STM32WBx5 Nucleo board to the host computer using the USB_USER connector.

Make sure that the 5 V sources jumper connector is plugged into the USB MCU.

6.2.2 Wireshark

Install Wireshark v2.4.6 or later available from http://www.wireshark.org and add installation path to the path environment variable if it is not already done.

Once done, the user must copy the python sniffer script stm32cubeMonRf_sniffer.py and the associated stm32cubeMonRf_sniffer.bat file in the Wireshark extcap directory. Files are available in the sniffer directory where the tool is installed, by default for Windows: *\Program Files (x86)\STMicroelectronics\STM32CubeMonitor-RF\sniffer*.

Wireshark extcap path is available in the Help/About Wireshark menu under the Folders tab.

Note that for macOS and Linux, the *stm32cubeMonRf_sniffer.py* file must have the execute permission.

6.2.3 Python

Install python v2.7.x or later available from https://www.python.org/downloads and add installation path to the path environment variable if it's not already done.

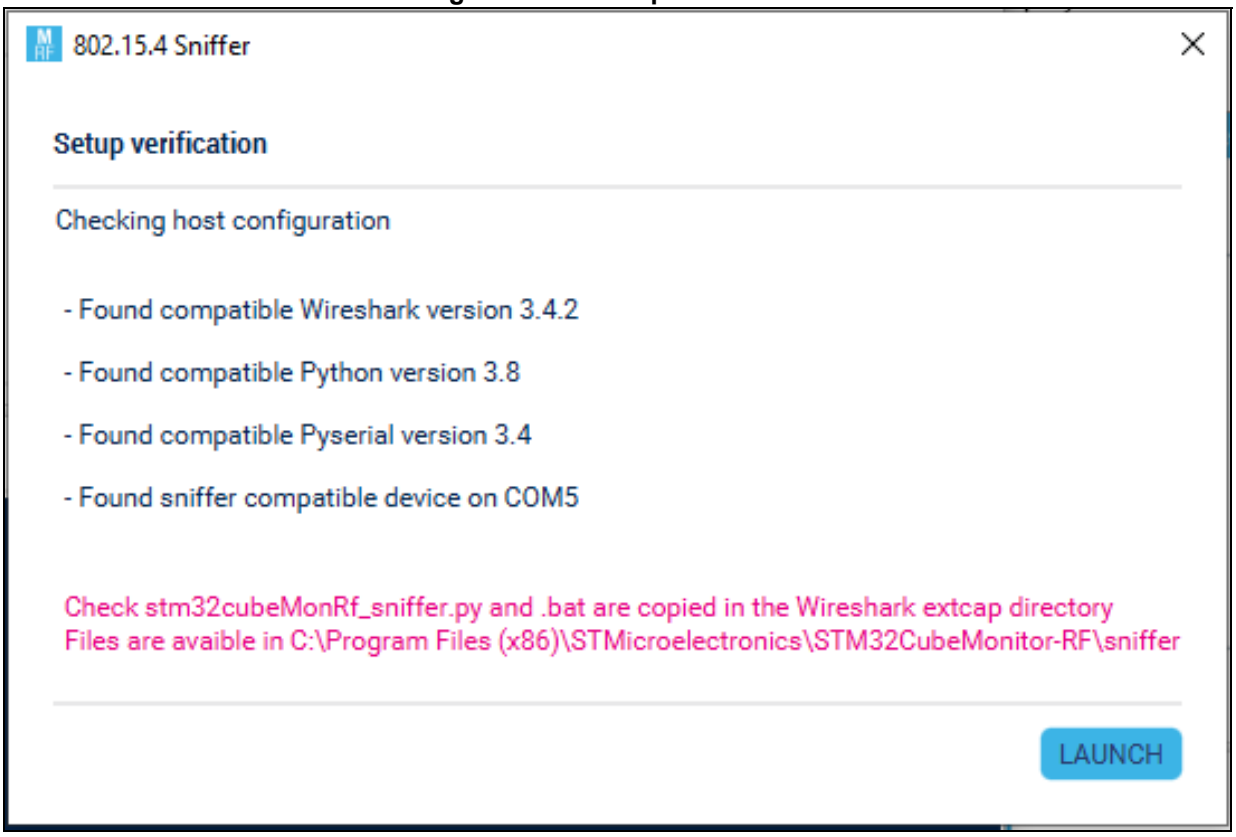
The user also needs to install the python serial port extension, pyserial, available from https://pypi.org/project/pyserial.

6.3 Setup verification

The sniffer can be invoked using the 802.15.4 SNIFFER button available on the welcome screen or through the Settings/mode menu available in the menu bar. In both cases, the tool checks that the prerequisites are fulfilled.

If it is not the case, the user is asked to correct it. Otherwise following pop-up window is displayed. To launch the sniffer, click on the LAUNCH button.

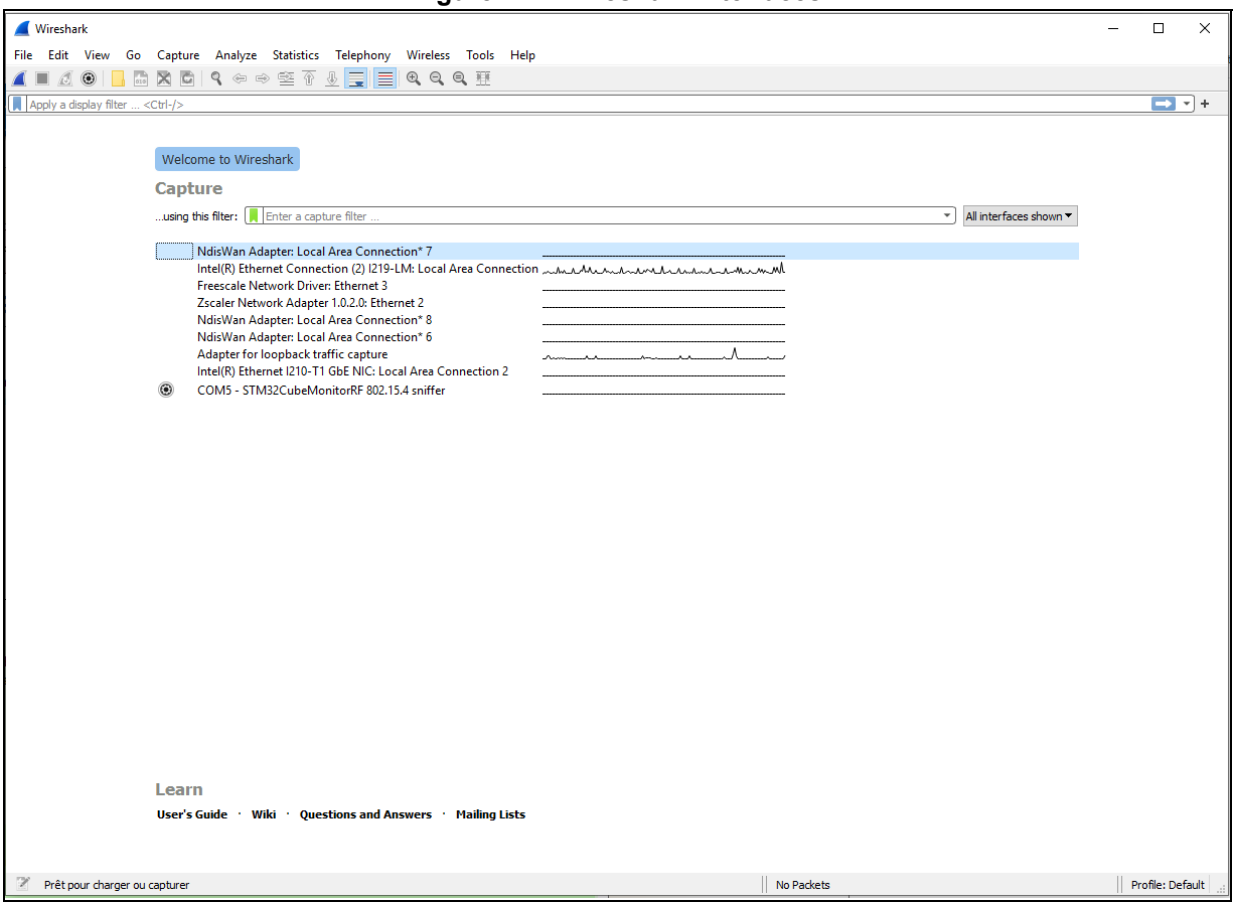
Figure 141. Prerequisite check



6.3.1 Sniffer Launch

Once Wireshark is launched, the user is proposed to choose the interface to sniff.

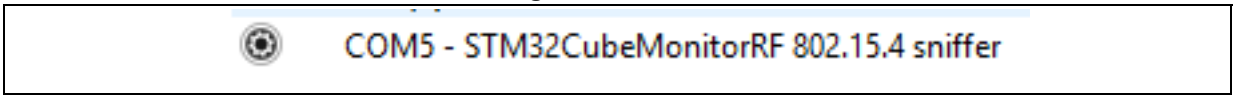
Figure 142. Wireshark interfaces



6.3.2 Select interface

Choose the interface corresponding to the device configured for sniffing by clicking on the wheel.

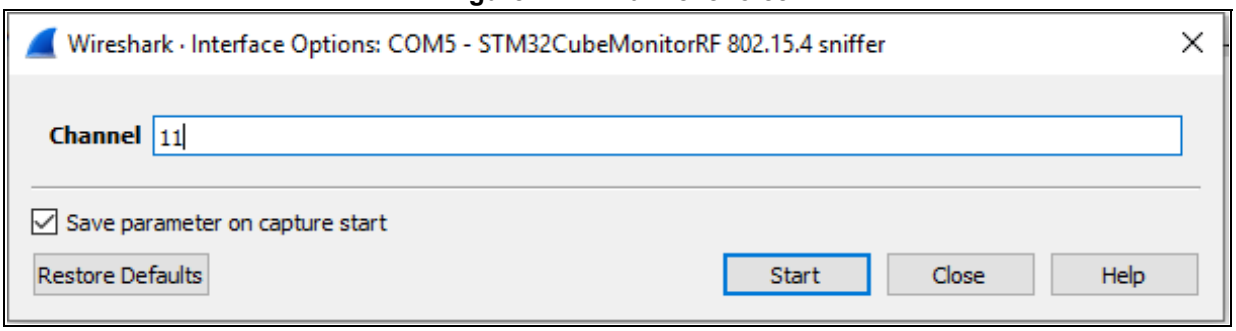
Figure 143. Wheel



6.3.3 Configure channel.

The user is asked to choose the channel to be sniffed.

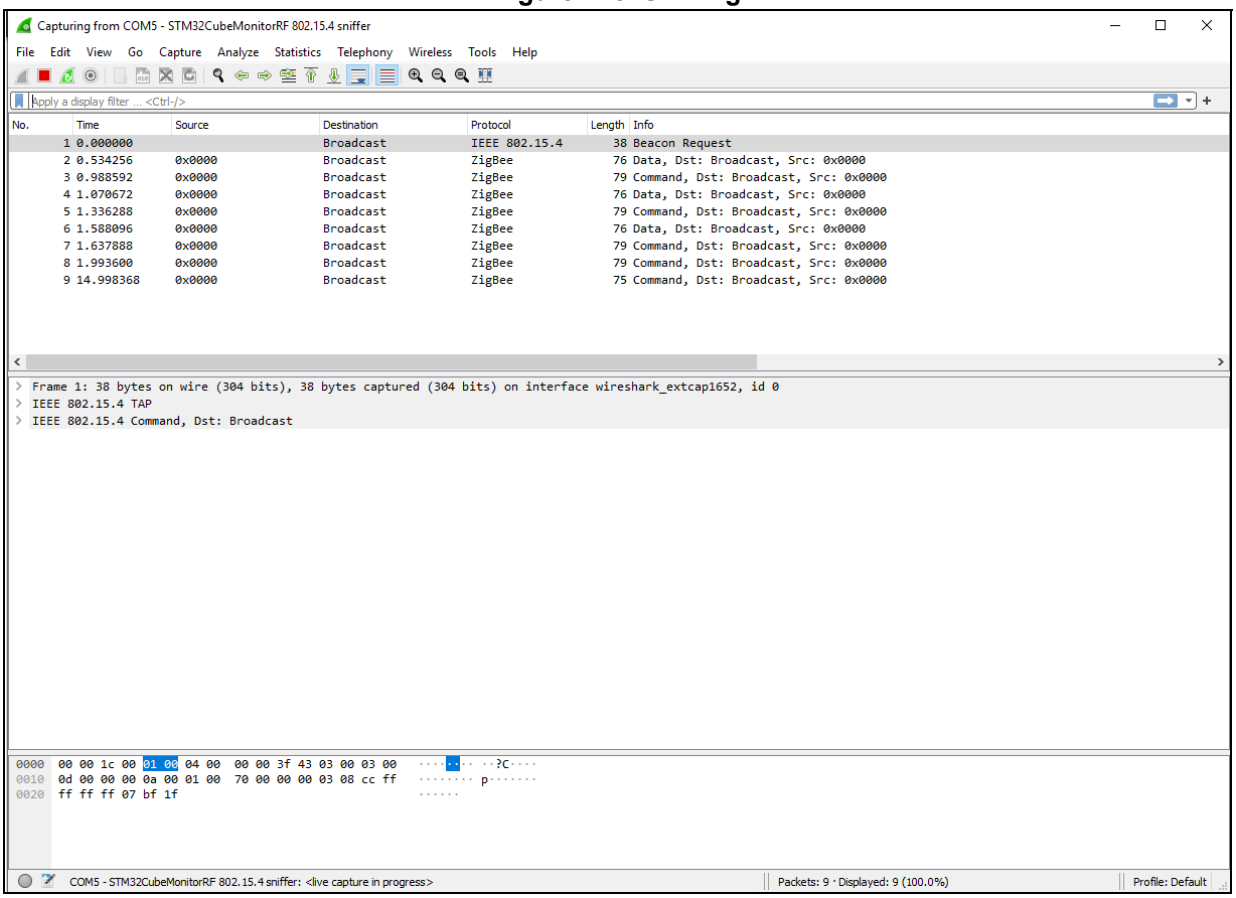
Figure 144. Channel choice



6.3.4 Start sniffing.

Once the channel is selected, click on start. Sniffed packets list appears at the top of the tool, details of the selected packet in the middle, and packet byte in the bottom.

Figure 145. Sniffing



Appendix A Beacon configuration format

The beacon configuration file is binary. Its content is explained in *Table 4*.

Table 4. Beacon configuration format

Byte #	Name	Value	Description
0	Address type	0,1,2	Address type: 0 = board default address 1 = random address 2 = static address provided in the block
1 - 6	Address	address	static address for the beacon. Valid only if the address type is 2.
7	Tx power	*0x00-0x1F*	Tx power to be used for the beacon. Value *PA_Level* of command ACI_HAL_SET_TX_POWER_LEVEL: 0 to 31
8	Beacon additional feature	0 or 1	0: No additional feature 1: TLM activated Other values reserved
9	Advertising payload length	13-32	Length of payload data
10 - 41	Advertising payload	-	Beacon advertisement payload, to be inserted in the advertisement

Revision history

Table 5. Document revision history

Date	Revision	Changes
27-Nov-2017	1	Initial version
25-Jan-2018	2	Updated: – *Introduction* – *Section 3.3.2: Test mode receiver (RX)* Added: – Two tables: *Table 2: Specific AD encoding for code example* and *Table 3: Search filtering* – Twelve new figures – *Section 3.2.1: How to send an ACI command* – *Section 3.2.2: Search function* – *Section 3.4.4: Script report* – *Section: Pause command in the script* – *Section 3.5.3: Advertising change for OTA in ST example*
23-Aug-2018	3	Complete content reorganized to explain tool support to the original *Bluetooth low energy mode* in *Section 3* and the new *OpenThread mode* in *Section 4*.
13-Feb-2019	4	Updated: – *Section 4: OpenThread mode* and most of the figures with new version tool Added: – *Section 5: 802.15.4 RF test mode*
12-Jul-2019	5	Updated: – Tool version 2.4.0 – *Section 3.5: OTA transfer* simplified. Details are reported in the application note.
30-Mar-2020	6	Added: – Support of STM32WB35 with updated paths – *Section 4.4* on Thread network exploration feature
12-Nov-2020	7	Updated: – Tool version 2.6.0 Added: Six new sections: – *Section 5.2.1: Frame mode* – *Section 5.2.2: Continuous modulated mode* – *Section 5.3.1: Packet error rate (PER) test* – *Section 5.3.2: Link quality assessment (LQI) test* – *Section 5.3.3: Energy detection (ED) test* – *Section 5.3.4: Channel clear assessment (CCA) test* dealing with all the applicable tests in the *Transmitter test (TX) mode* and *Receiver test (RX) mode*

Table 5. Document revision history (continued)

Date	Revision	Changes
8-Feb-2021	8	All modifications linked to new feature 802.15.4 sniffer Updated: – *Introduction* – *Section 1.2: Welcome screen* with *Figure 1* – *Section 2.2.3: VCP device* – *Figure 61*, *Figure 62*, *Figure 65*, *Figure 105*, and *Figure 114* Added: – *Section 6: 802.15.4 Sniffer* with *Figure 141* to *Figure 145*
22-Jul-2021	9	All modifications linked to new tool version 2.8.0 Updated: – *Section 2.2: VCOM / UART connection* – *Section 3.3.3: PER* – *Section 3.4.3: Scripts modification* – *Section 3.5.3: Use the tool to perform an OTA update* – *Section 3.6.3: Configuration of the beacon with STM32CubeMonitor-RF* – *Figure 61*, *Figure 62*, *Figure 65*, and *Figure 80* to *Figure 82*