Introduction

STM32CubeMonitor-RF (STM32CubeMonRF) is a software tool, which helps the 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 packet error rate (PER) on BLE and 802.15.4 technologies.
- It sends commands to a Bluetooth® low energy (BLE) controller for standardized tests.
- It sends and receives BLE commands for fast application prototyping.
- It configures a variety of beacons via BLE commands.
- It transfers data over-the-air (OTA) from one device to another, in order to configure or program a remote device without wired connection.
- It sends commands to an OpenThread device for application prototyping.
## 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 .............................................. 18  
   2.3 Opening COM ................................................ 19  

3. **BLE mode** .................................................... 21  
   3.1 Presentation .................................................. 21  
      3.1.1 Panels ................................................... 21  
   3.2 ACI Commands panel .......................................... 22  
      3.2.1 How to send an ACI command ............................. 22  
      3.2.2 Search function .......................................... 23  
      3.2.3 Filter usage .............................................. 23  
      3.2.4 How to fill parameters. Fixed field / editable field 23  
      3.2.5 Log functionalities ....................................... 24  
   3.3 RF tests panel ............................................... 27  
      3.3.1 Test mode transmitter (TX) .............................. 28  
      3.3.2 Test mode receiver (RX) ................................ 31  
      3.3.3 PER ..................................................... 34  
   3.4 Scripts ....................................................... 39  
      3.4.1 Launching scripts ....................................... 40  
      3.4.2 Script recording ......................................... 41
3.4.3 Scripts modification ........................................... 42
3.4.4 Script report .................................................... 42
3.4.5 List of scripts commands ..................................... 44
3.5 OTA transfer ....................................................... 47
  3.5.1 OTA presentation ............................................. 47
  3.5.2 OTA procedure ............................................... 49
  3.5.3 Advertising change for OTA in ST example ............... 50
  3.5.4 Use the tool to perform OTA update ....................... 50
3.6 Beacon ............................................................ 54
  3.6.1 Beacon presentation ......................................... 54
  3.6.2 Beacon configuration methods: ............................ 55
  3.6.3 Configuration of beacon with STM32CubeMonitor-RF .... 57
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 .................................................. 72
  4.3 OpenThread scripts tab ....................................... 74
    4.3.1 OpenThread script example .............................. 76
    4.3.2 List of script commands ................................. 76

5 802.15.4 RF test mode .............................................. 78
  5.1 Presentation .................................................... 78
  5.2 Transmitter (TX) test mode .................................. 79
  5.3 Receiver test (RX) mode ...................................... 79
  5.4 Packet error rate (PER) mode ................................ 80
    5.4.1 Connecting the additional device to play the role of packet generator (tester). ......................... 81
    5.4.2 Configure the parameters of tester. ................. 82
    5.4.3 Configure the parameters of device under test (DUT). ......................... 83
    5.4.4 Configure the measurement. ............................. 84

Appendix A Beacon configuration format .......................... 87
List of tables

Table 1. OTA loader address table ................................................................. 48
Table 2. Specific AD encoding for code example ............................................ 50
Table 3. Search filtering .............................................................................. 51
Table 4. Measurement setting ...................................................................... 85
Table 5. Beacon configuration format ............................................................. 87
Table 6. Document revision history ............................................................... 88
# List of figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1.</td>
<td>Welcome screen</td>
<td>10</td>
</tr>
<tr>
<td>Figure 2.</td>
<td>Main screen</td>
<td>11</td>
</tr>
<tr>
<td>Figure 3.</td>
<td>Menu bar</td>
<td>11</td>
</tr>
<tr>
<td>Figure 4.</td>
<td>Connection bar</td>
<td>12</td>
</tr>
<tr>
<td>Figure 5.</td>
<td>ACI Commands panel</td>
<td>13</td>
</tr>
<tr>
<td>Figure 6.</td>
<td>Log area</td>
<td>14</td>
</tr>
<tr>
<td>Figure 7.</td>
<td>Typical connection with a Nucleo board</td>
<td>15</td>
</tr>
<tr>
<td>Figure 8.</td>
<td>Connection with a remote device</td>
<td>15</td>
</tr>
<tr>
<td>Figure 9.</td>
<td>Connection with a second device</td>
<td>16</td>
</tr>
<tr>
<td>Figure 10.</td>
<td>VCOM connection BLE</td>
<td>16</td>
</tr>
<tr>
<td>Figure 11.</td>
<td>VCOM connection Thread</td>
<td>17</td>
</tr>
<tr>
<td>Figure 12.</td>
<td>UART connection</td>
<td>18</td>
</tr>
<tr>
<td>Figure 13.</td>
<td>VCP connection</td>
<td>19</td>
</tr>
<tr>
<td>Figure 14.</td>
<td>Opening COM</td>
<td>19</td>
</tr>
<tr>
<td>Figure 15.</td>
<td>Successful COM</td>
<td>20</td>
</tr>
<tr>
<td>Figure 16.</td>
<td>ACI Commands panel</td>
<td>21</td>
</tr>
<tr>
<td>Figure 17.</td>
<td>How to send an ACI command</td>
<td>22</td>
</tr>
<tr>
<td>Figure 18.</td>
<td>Search button</td>
<td>23</td>
</tr>
<tr>
<td>Figure 19.</td>
<td>Fixed parameter</td>
<td>23</td>
</tr>
<tr>
<td>Figure 20.</td>
<td>Editable parameter</td>
<td>24</td>
</tr>
<tr>
<td>Figure 21.</td>
<td>Predefined values</td>
<td>24</td>
</tr>
<tr>
<td>Figure 22.</td>
<td>Help details</td>
<td>24</td>
</tr>
<tr>
<td>Figure 23.</td>
<td>Log functionalities</td>
<td>25</td>
</tr>
<tr>
<td>Figure 24.</td>
<td>More button</td>
<td>25</td>
</tr>
<tr>
<td>Figure 25.</td>
<td>Message details</td>
<td>26</td>
</tr>
<tr>
<td>Figure 26.</td>
<td>Purple error messages</td>
<td>26</td>
</tr>
<tr>
<td>Figure 27.</td>
<td>Gray second board messages</td>
<td>27</td>
</tr>
<tr>
<td>Figure 28.</td>
<td>Test mode selection</td>
<td>27</td>
</tr>
<tr>
<td>Figure 29.</td>
<td>Change test mode</td>
<td>28</td>
</tr>
<tr>
<td>Figure 30.</td>
<td>Select test mode</td>
<td>28</td>
</tr>
<tr>
<td>Figure 31.</td>
<td>Test mode transmitter</td>
<td>29</td>
</tr>
<tr>
<td>Figure 32.</td>
<td>Transmitting message</td>
<td>30</td>
</tr>
<tr>
<td>Figure 33.</td>
<td>Transmitted packets count</td>
<td>30</td>
</tr>
<tr>
<td>Figure 34.</td>
<td>Test mode receiver</td>
<td>31</td>
</tr>
<tr>
<td>Figure 35.</td>
<td>RSSI measurement</td>
<td>32</td>
</tr>
<tr>
<td>Figure 36.</td>
<td>RSSI measurement graph</td>
<td>33</td>
</tr>
<tr>
<td>Figure 37.</td>
<td>RF RSSI measurement big display</td>
<td>34</td>
</tr>
<tr>
<td>Figure 38.</td>
<td>PER definition</td>
<td>34</td>
</tr>
<tr>
<td>Figure 39.</td>
<td>PER tester connection</td>
<td>35</td>
</tr>
<tr>
<td>Figure 40.</td>
<td>PER tester connected</td>
<td>36</td>
</tr>
<tr>
<td>Figure 41.</td>
<td>PER tester configuration</td>
<td>36</td>
</tr>
<tr>
<td>Figure 42.</td>
<td>DUT configuration</td>
<td>37</td>
</tr>
<tr>
<td>Figure 43.</td>
<td>PER test parameters</td>
<td>38</td>
</tr>
<tr>
<td>Figure 44.</td>
<td>PER and RSSI measurement graph</td>
<td>39</td>
</tr>
<tr>
<td>Figure 45.</td>
<td>Launching scripts</td>
<td>40</td>
</tr>
<tr>
<td>Figure 46.</td>
<td>Script execution</td>
<td>41</td>
</tr>
<tr>
<td>Figure 47.</td>
<td>Script recording buttons</td>
<td>41</td>
</tr>
<tr>
<td>Figure 48.</td>
<td>Sample script</td>
<td>42</td>
</tr>
<tr>
<td>Figure 49.</td>
<td>Script report</td>
<td>43</td>
</tr>
<tr>
<td>Figure 50.</td>
<td>Script verdict</td>
<td>43</td>
</tr>
<tr>
<td>Figure 51.</td>
<td>Script pause</td>
<td>44</td>
</tr>
<tr>
<td>Figure 52.</td>
<td>Example</td>
<td>45</td>
</tr>
<tr>
<td>Figure 53.</td>
<td>Loop simple example</td>
<td>45</td>
</tr>
<tr>
<td>Figure 54.</td>
<td>Loop second simple example</td>
<td>45</td>
</tr>
<tr>
<td>Figure 55.</td>
<td>Loop decrement</td>
<td>46</td>
</tr>
<tr>
<td>Figure 56.</td>
<td>Loop specific increment</td>
<td>46</td>
</tr>
<tr>
<td>Figure 57.</td>
<td>Nested loop</td>
<td>46</td>
</tr>
<tr>
<td>Figure 58.</td>
<td>Loop script verdict example</td>
<td>47</td>
</tr>
<tr>
<td>Figure 59.</td>
<td>Loop script verdict display</td>
<td>47</td>
</tr>
<tr>
<td>Figure 60.</td>
<td>Script loop error</td>
<td>47</td>
</tr>
<tr>
<td>Figure 61.</td>
<td>Search procedure</td>
<td>51</td>
</tr>
<tr>
<td>Figure 62.</td>
<td>Scanning</td>
<td>52</td>
</tr>
<tr>
<td>Figure 63.</td>
<td>No device found</td>
<td>52</td>
</tr>
<tr>
<td>Figure 64.</td>
<td>Device found</td>
<td>52</td>
</tr>
<tr>
<td>Figure 65.</td>
<td>Select the device and parameters</td>
<td>53</td>
</tr>
<tr>
<td>Figure 66.</td>
<td>Configuring in OTA</td>
<td>53</td>
</tr>
<tr>
<td>Figure 67.</td>
<td>Progress bar</td>
<td>54</td>
</tr>
<tr>
<td>Figure 68.</td>
<td>Beacon presentation</td>
<td>54</td>
</tr>
<tr>
<td>Figure 69.</td>
<td>Beacon usage</td>
<td>54</td>
</tr>
<tr>
<td>Figure 70.</td>
<td>Online beacon</td>
<td>55</td>
</tr>
<tr>
<td>Figure 71.</td>
<td>Offline beacon</td>
<td>56</td>
</tr>
<tr>
<td>Figure 72.</td>
<td>Beacon over-the-air configuration</td>
<td>56</td>
</tr>
<tr>
<td>Figure 73.</td>
<td>Selecting the beacon mode</td>
<td>57</td>
</tr>
<tr>
<td>Figure 74.</td>
<td>Common parameters</td>
<td>57</td>
</tr>
<tr>
<td>Figure 75.</td>
<td>iBeacon parameters</td>
<td>58</td>
</tr>
<tr>
<td>Figure 76.</td>
<td>Eddystone UID</td>
<td>59</td>
</tr>
<tr>
<td>Figure 77.</td>
<td>Eddystone URL</td>
<td>59</td>
</tr>
<tr>
<td>Figure 78.</td>
<td>Online mode transfer configuration</td>
<td>60</td>
</tr>
<tr>
<td>Figure 79.</td>
<td>Offline mode transfer configuration</td>
<td>60</td>
</tr>
<tr>
<td>Figure 80.</td>
<td>OTA mode transfer configuration scan</td>
<td>60</td>
</tr>
<tr>
<td>Figure 81.</td>
<td>OTA mode transfer configuration select device</td>
<td>61</td>
</tr>
<tr>
<td>Figure 82.</td>
<td>OTA mode transfer configuration transmission</td>
<td>61</td>
</tr>
<tr>
<td>Figure 83.</td>
<td>ACI Utilities panel</td>
<td>62</td>
</tr>
<tr>
<td>Figure 84.</td>
<td>Select checkbox</td>
<td>62</td>
</tr>
<tr>
<td>Figure 85.</td>
<td>Scan parameters</td>
<td>63</td>
</tr>
<tr>
<td>Figure 86.</td>
<td>Scanning</td>
<td>64</td>
</tr>
<tr>
<td>Figure 87.</td>
<td>Select device</td>
<td>64</td>
</tr>
<tr>
<td>Figure 88.</td>
<td>Back</td>
<td>64</td>
</tr>
<tr>
<td>Figure 89.</td>
<td>Init</td>
<td>64</td>
</tr>
<tr>
<td>Figure 90.</td>
<td>Connecting</td>
<td>65</td>
</tr>
<tr>
<td>Figure 91.</td>
<td>Connection error</td>
<td>65</td>
</tr>
<tr>
<td>Figure 92.</td>
<td>Connected icon</td>
<td>65</td>
</tr>
<tr>
<td>Figure 93.</td>
<td>Services list</td>
<td>65</td>
</tr>
<tr>
<td>Figure 94.</td>
<td>Characteristics list</td>
<td>66</td>
</tr>
<tr>
<td>Figure 95.</td>
<td>Read value</td>
<td>66</td>
</tr>
<tr>
<td>Figure 96.</td>
<td>Write value</td>
<td>67</td>
</tr>
<tr>
<td>Figure 97.</td>
<td>Indicate value changed</td>
<td>67</td>
</tr>
<tr>
<td>Figure 98.</td>
<td>Notify value changed</td>
<td>68</td>
</tr>
<tr>
<td>Figure 99.</td>
<td>Notifying</td>
<td>68</td>
</tr>
<tr>
<td>Figure 100.</td>
<td>Advertising parameters</td>
<td>69</td>
</tr>
</tbody>
</table>
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 is applicable to STM32CubeMonitor-RF version 2.3.0 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

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

The check box ‘Remember my choice’ memorizes the selection, so that the next application launch directly opens it, without welcome screen.
1.3 **Main screen**

The main screen of the tools is subdivided in 4 parts: menu bar, connection bar, panels and log area.

![Main screen](image)

**Figure 2. Main screen**

1.3.1 **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 connected board.

The ‘Help’ menu provides information about the version of 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 web site
- The STMicroelectronics icon leads to the STMicroelectronics web site.

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 BLE wireless stack. When many tests are performed, the button should be used to reset the stack at start of each test.

1.3.3 Panels

The panels are used to perform specific operations. Each panel groups different functions.

The ‘ACI Commands’ panel example is illustrated in Figure 5.

Each panel is detailed in a specific section of the document:

- **Section 3.2: ACI Commands panel on page 22**
- **Section 3.4: Scripts on page 39**
- **Section 3.6: Beacon on page 54**
- **Section 3.3: RF tests panel on page 27**
- **Section 3.7: ACI utilities on page 62**
1.3.4 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 24.
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 ‘main device’. This is the board that the user wants to exercise with the tool. This board is also named device under test (DUT).

**Figure 7. Typical connection with a Nucleo board**

Some operation, like download over-the-air, involves communication with another device. This other device is referred to as ‘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 '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 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 BLE applications, CLI 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 BLE**

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 BLE, a special firmware in STM32WBxx called ‘transparent mode’ copies the data received on the Rx pin to the BLE stack. Data sent back by the BLE stack follows the reverse path.

The transparent mode firmwares are available in the STM32CubeWB Firmware Package (see folder \Projects\NUCLEO-WB55.Nucleo\Applications\BLE\ble_transparent_mode).

The wireless stack firmware stm32wb5x_BLE_Stack_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 firmwares are available in the STM32CubeWB Firmware Package (see folder \Projects\NUCLEO-WB55.Nucleo\Applications\Thread\Thread_Cli_Cmd).

The wireless stack firmware stm32wb5x_Thread_FTD_fw.bin is available in \Projects\STM32WB_Copro_Wireless_Binaries.

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.

The Cli_Phy_802_15_4_M4.bin firmware binary is provided in the tool folder:
- for Windows® <Documents>\STMicroelectronics\STM32CubeMonitor-RF\firmwares
- for Linux®, <userhome>\STMicroelectronics\STM32CubeMonitor-RF\firmwares
- for macOS® it is inside the document folder provided in the setup package

The wireless stack firmware stm32wb5x_rfmonitor_phy802_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 automatic driver setup, the user must install the VCOM driver manually.

---

**Figure 11. VCOM connection Thread**

![Diagram of VCOM connection Thread](image-url)
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 802.15.4 test:
1. Flash the Nucleo binary Cli_Phy_802_15_4_M4.bin and the wireless stack stm32wb5x_rfmonitor_phy802_15_4_fw.bin in the dongle (with DFU).
2. Move solder bridge SB2 to SB6 (connection of PB7 to CN2.7).
3. Connect 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.
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.

1. For BLE:
   the firmware is in \Projects\NUCLEO WB55.USBDongle\Applications\BLE\ble_transparent_mode_vcp.
   The wireless stack is in \Projects\STM32WB_Copro_Wireless_Binaries\stm32wb5x_BLE_Stack_fw.bin

2. For Thread:
   the firmware is in \Projects\NUCLEO-WB55.USBDongle\Applications\Thread\Thread_Cmd_Cmd.
   The wireless stack is in \Projects\STM32WB_Copro_Wireless_Binaries\stm32wb5x_Thread_FTD_fw.bin

### 2.3 Opening COM

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

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 device, drivers need to be installed manually.
- Select the serial port to use in the picklist (Comx on Windows, ttyACMx on Linux and Mac).
- Click CONNECT

The board is connected, and the version is displayed in the right side of the bar.
When the CONNECT button is pressed, the software attempts to communicate with the device to read the firmware and HW versions. If the connection is not working, the tool displays an error and disconnects the COM port.

**Caution:** In case of 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 driver may not install automatically. If the tool is not showing the COM port in the list, please 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 during a few seconds, and STM32CubeMonitor-RF is unable to connect.
  - The user must wait the end of modemmanager activity before opening the COM port.
  - If 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 dialout 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 BLE mode

3.1 Presentation

3.1.1 Panels

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

Figure 16. ACI Commands panel

The main panels are: ACI Commands, Scripts, Beacon, RF Tests, 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 BLE stack. Commands are grouped by categories. These commands allow the user to configure the BLE stack and activate the communication with remote devices.

3.2.1 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 the parameters of the command. Default values are used otherwise.
3. Click on ‘SEND COMMAND’ The command is sent to 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 character is entered, matching commands are filtered in the list. The match may be any part of the command name, it is not necessary to start from the beginning.
- Click once on the command to select it (do not use double click).

![Figure 18. Search button](image)

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 to see 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 value and are not editable, while others are totally free or take only some values. The tool guides the user to fill the parameters:

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

![Figure 19. Fixed parameter](image)

- Editable parameter: editable parameter is surrounded by a blue rounded box. Value is editable inside the field. Edit is blocked if value is too long for the field.
### 3.2.5 Log functionalities

The log area is located in the right part of the tool. 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 command complete.

The commands with ‘Command Status’ usually have other events coming later. This events are also displayed in the log area.

Some asynchronous event may come from the device, and is also displayed in this area.

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

Sometimes, all the information of a message does not fit in the area used for log. The button *Figure 24* opens a new window showing the message details:
The details show all decoded message parameters. The ‘Literal’ column shows predefined text for the parameter values (Op-Code and others). The ‘Info’ column provides some description of 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 in other windows.

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

**Color code**

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

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

Log on a dark gray background are 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.
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 log history, untick the box to disable 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 tests panel
The RF tests panel is used to perform the radio-frequency tests on the main device. The RF tests are grouped in 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.

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 test mode**

- Click on ‘test mode’ in the top bar.

**Figure 30. Select test mode**

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

### 3.3.1 Test mode transmitter (TX)

The TX mode is used to set the BLE transmitter in emission. Two transmission modes are defined: transmission of data, or emission of tone.
Tone generation

The tone generation performs the emission of a continuous sin 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. Advertising channel index do not follow the frequency order. Channels 37, 38 and 39 are the advertising channels. See BLUETOOTH SPECIFICATION Version 4.2 [Vol 6, Part B] ch1.4.1 for details.
4. Select the PHY modulation to use (The modulation not supported by the device are not listed).
5. Click on ‘START TONE’ button.

The emission starts, ‘START TONE’ button is changed to ‘STOP TONE’, and Transmitting information is displayed:
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 payload are available:

- Pseudo-Random bit sequence 9 (PRBS9)
- Pattern of alternating bits ‘11110000’
- Pattern of alternating bits ‘10101010’
- Pseudo-Random bit sequence 15 (PRBS15)
- Pattern of All ‘1’ bits
- Pattern of All ‘0’ bits
- Pattern of alternating bits ‘00001111’
- Pattern of alternating bits ‘01010101’

The sequence length is defined by the ‘Length of data’ picklist. This is the length of 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, start button is changed to ‘STOP TX’, and Transmitting is displayed. The sequence is repeated until 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.

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 (see 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

![Test mode receiver](image)

**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 packet received 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 (see Section 3.3.3: PER for details).

If the ‘Get RSSI’ check box is selected, the tool performs RSSI measurement.

**RSSI measurement:**

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

When 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

#### Receiver

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>RX Frequency</td>
<td>2402 MHz (Channel 37)</td>
</tr>
<tr>
<td>PHY</td>
<td>0x01 - Receiver act to use the LE 1M PHY</td>
</tr>
<tr>
<td>Index modulation</td>
<td>0x00 - Assume transmitter will have a standard modulation index</td>
</tr>
<tr>
<td>Get RSSI</td>
<td>Measurement period (sec): 3</td>
</tr>
</tbody>
</table>

**Test measurement at 2402 MHz (Channel 37)**

- Transmitted packets count: [Count]
- Received packets count: [Count]
- Receiving...
- Packet Error Rate (PER): [Count]
- RSSI: -10.00 dBm
Figure 36. RSSI measurement graph

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.
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 transmitter to 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](image)

\[
\text{PER} = 100 \times \frac{Ntx - Nrx}{Ntx} \%
\]

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 receiver, and depends on parameters like 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 know performances in the setup.

**PER test mode**

The tool provides a special test mode dedicated to 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 test the DUT (tester)

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

**Figure 39. PER tester connection**

- Plug the device in the computer (same requirements as first device, see *Chapter 2.2*).
- Select the serial port to use in the picklist.
- Click on the ‘CONNECT’ button.
The board information is displayed on the right.

When 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:

- Select the TX power level with the picklist.
- Select the transmission frequency with ‘TX Frequency’ picklist.
- Select the length of packet to send (same as TX test).
- Select the content of packet payload (same as TX test).
- 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 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 add some RSSI measurement between each PER measurement. When it is activated, the tool performs a PER test for the Measurement period, compute PER, and then make a 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. Report is saved at the end of the tests.

When option have been configured, click on ‘START TEST’ button:

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

Button is changed to ‘STOP’ and ‘Testing…’ is displayed.

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 bars 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 enter file name directly.
- Click on ‘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.
- Script is manually stopped with the ‘Stop script’ button.
Script examples are provided with the tool (Sample script, loop, beacon creation, etc).

For Windows® scripts are in 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 is directly recorded in a script. Some script buttons are located at the bottom of ACI panel:

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

At the end of recording, click the ‘Stop’ button. The tool asks the script name before saving.
3.4.3 Scripts modification

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. Line starts with the command name, followed by parameters in brackets, separated by 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.
The result is stored in a new file, in same path as script, with a name in the form: ‘verdict_SampleScript_18-12-2017_17-27-52’. The name is built with concatenation of:

- ‘verdict_’
- the script name
- the current date
- the current time
- .txt extension

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

The ‘ACI status’ column, have the status of ACI response. 0x00 is a success status, other values are errors.

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

If the user presses ‘yes’, report is generated in the folder of current script. If the user presses ‘no’, report is not saved.
If the tick box ‘Generate report’ in not checked, no report is generated at the end of 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 or the performance are not verified, there is no PASS/FAIL criteria on the results.

3.4.5 List of scripts 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 of 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 in 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

![Script pause](image)

The ‘OK’ button allows to continue the script.

Command: Pause ('User comment')

The user text must be enclosed between quote marks (").
Figure 52. Example

```plaintext
# 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 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 the end value. In this example, the counter count is be 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 on 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' 3 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 value. In this case, add an ampersand (&) after the first bracket. The tool replaces the counter name by 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**

```plaintext
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**

```plaintext
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 different counter name.

**Figure 57. Nested loop**

```plaintext
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

<table>
<thead>
<tr>
<th>Loop(FREQ,13,15)</th>
</tr>
</thead>
<tbody>
<tr>
<td>EndLoop</td>
</tr>
</tbody>
</table>

Figure 59. Loop script verdict display

<table>
<thead>
<tr>
<th>-- Loop Start (FREQ=13) --</th>
<th>--</th>
<th>--</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>-- Loop (FREQ=14) --</td>
<td>--</td>
<td>--</td>
<td></td>
</tr>
<tr>
<td>-- Loop (FREQ=15) --</td>
<td>--</td>
<td>--</td>
<td></td>
</tr>
<tr>
<td>-- Loop End (FREQ) --</td>
<td>--</td>
<td>--</td>
<td></td>
</tr>
</tbody>
</table>

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

In case of 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 cable. The data are applicative data, like user configuration, picture, music or firmware. STM32CubeMonitor-RF provides a transfer function from the computer to the remote device over BLE.

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 the customer requirements.

It is not possible to load the BLE stack.

## OTA loader

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

- If the OTA boot mode is not requested, and Flash is not empty, the loader jumps to the firmware. The user firmware is stored in Flash at address 0x7000.
- If OTA boot has been requested or Flash is empty at 0x7000, the OTA loader erases the required Flash area to prepare the transfer, then starts advertising of device as an OTA device and wait connections.

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

Service Name: ST OTA, Service UUID 0000FE20-cc7a-482a-984a-7f2ed5b3e58f

Characteristics and associated UUID:

- Base address: 0000FE22-8e22-4541-9d4c-21edae82ed19 (Write)
- File Upload confirmation - Reboot: 0000FE23-8e22-4541-9d4c-21edae82ed19 (notify)
- Raw data: 0000FE24-8e22-4541-9d4c-21edae82ed19 (write)

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>
<thead>
<tr>
<th>Flash address:</th>
<th>Flash content:</th>
</tr>
</thead>
<tbody>
<tr>
<td>+0x0000</td>
<td>OTA bootloader</td>
</tr>
<tr>
<td>+0x1000</td>
<td>OTA bootloader</td>
</tr>
<tr>
<td>+0x6000</td>
<td>Free for user data</td>
</tr>
<tr>
<td>+0x7000</td>
<td>User application</td>
</tr>
<tr>
<td>+0x8000</td>
<td>User application</td>
</tr>
<tr>
<td>+0x….</td>
<td>User application</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Flash address:</th>
<th>Flash content:</th>
</tr>
</thead>
<tbody>
<tr>
<td>+0x0000</td>
<td>OTA bootloader</td>
</tr>
<tr>
<td>+0x1000</td>
<td>OTA bootloader</td>
</tr>
<tr>
<td>+0x6000</td>
<td>Free for user data</td>
</tr>
<tr>
<td>+0x7000</td>
<td>User application</td>
</tr>
<tr>
<td>+0x8000</td>
<td>User application</td>
</tr>
<tr>
<td>+0x….</td>
<td>User application</td>
</tr>
</tbody>
</table>

- In the STM32WB55 sample code, the user data are transferred in the 0x6000 Flash area and the user program uses this data.
- For firmware update, the binary is stored at address 0x7000, and bootloader starts at this address after upload.
OTA enabled device

When the user application is running in normal mode, the OTA loader is not active, and the OTA service is not available. In order to perform OTA transfer, the application needs to reboot the device in OTA mode.

The firmware decides to reboot to OTA mode when a button on the board is pressed, or when any defined event occurs on the device.

Another option is to include an OTA reboot characteristic in the BLE profile used by the application. It is possible to use this characteristic to inform application that OTA is requested, and the application stops and reboots in OTA mode. This solution is implemented in the example delivered by ST.

The OTA reboot characteristic is defined in the documents:
- Name: ‘Request’
- UUID 0000FE11-8e22-4541-9d4c-21edae82ed19
- WriteWithoutResponse, 3 bytes, 0x01aabb, with aa = the sector number where erase starts, and bb = the number of sectors to erase.

When an OTA reboot is requested, the OTA loader increases the BLE address of 1 after reboot. This procedure force devices with cache to update the data for the new address.

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 activation of the OTA mode must be done according to the procedure defined by the user application:
- If the restart is done with a button or another user action, the firmware must set information in memory to indicate that OTA boot is requested, and indicate which part of Flash must be erased.
- If the action is done by writing the OTA reboot characteristic, the source device writes the characteristic with the area to erase information. Firmware stores in memory the part of Flash to be erased and triggers reset.

When OTA bootloader starts, the area defined before reboot is erased. Then the device is ready to start transfer.

Connect in OTA mode and transfer data

The source device first connects to the OTA loader and discover the details of service and characteristics to be able to transfer the data. Then the sequence is:
1. Configure the target device to send indication to the source device
2. Write in the target device the command to initiate the procedure, with exact storage address
3. Write each block of data. The blocks of data are 20-byte long, and the binary must be transferred in many blocks.
4. At the end of 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 Advertising change for OTA in ST example

The OTA procedure requires to find an OTA loader or OTA enabled device. This search may be done by connecting to every device found in the area but it is long and inefficient.

In order to have a faster connections, ST sample code adds an identification information in the advertising messages. With this information, a simple scan is sufficient to detect boards with OTA capabilities.

The information is added in an AD field of type FF (manufacturer specific):

<table>
<thead>
<tr>
<th>Table 2. Specific AD encoding for code example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Byte</td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
</tbody>
</table>

The OTA loader and OTA enabled device may be detected if the specific AD is found in the advertising, and bit for OTA Loader or OTA enabled device are set in the Feature group B bit field

3.5.4 Use the tool to perform OTA update

The OTA function is available in the device menu in the menu bar. Click on 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 BLE devices and list all the devices with OTA capabilities.
The tool provides an advertising filter to refine the search procedure with advertising message.

**Table 3. Search filtering**

<table>
<thead>
<tr>
<th>Filter</th>
<th>Search method</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>No filter</td>
<td>Scans all BLE devices and provides the list.</td>
<td>Some devices listed are not compatible for OTA.</td>
</tr>
<tr>
<td>Advertising filter</td>
<td>Scans all BLE devices, and provides a list of devices with ST OTA information.</td>
<td>Gives only the list of compatible devices.</td>
</tr>
</tbody>
</table>

To start search, click on the ‘SEARCH FOR DEVICES’ button.
The search procedure starts.

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

---

**Figure 62. Scanning**

[Image of the scanning interface]

---

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

---

**Figure 63. No device found**

[Image of the no device found interface]

---

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’.
The picklist displays the list of boards found:
- For device with BLE characteristic:
  - BLE address - Device name - OTA enabled
- For device already in OTA mode:
  - BLE address - Device name - OTA loader

The image base address is the place where the binary file must be stored on the target device. It is hexadecimal value, and must be multiple of 0x1000 to match with Flash sector.

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

The configuration is finished, 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

2. Second step, the transfer process to the OTA bootloader is performed. The data are transferred by block of 20 bytes. In order to avoid overload of log windows, the log
information related to block transfer are not displayed. Only the flow control event may be 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 do not receive any data.
The data shared by beacon are very small. This data are received by a connected device and application on the device are 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 web server more information about the beacon. Application gets information related to geographical position of 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 fix content, allowing to identify the beacon.
- Eddystone UID: this is defined by Google. The beacon transmits fixed content (UID), which is a unique ID, referenced in Google database to interact with applications.
- Eddystone URL: another Google format, provide a short URL, to use for ‘Physical web’.
- Eddystone TLM: an additional beacon advertising information providing beacon information (battery status, temperature).
- Eddystone EID: similar to UID, but broadcasts encrypted data to provide better security.

3.6.2 Beacon configuration methods:
STM32CubeMonitor-RF is used to generate and configure beacon. 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 advertising packet. The main device acts as a beacon as long as it is powered.

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 board is reset or powered off.

Offline beacon
The offline beacon mode is used to prepare the configuration of a board not directly connected to 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 target Flash at address 0x6000. The beacon firmware reads the data and configures the advertising block accordingly. Details of configuration file is described in Table 5: Beacon configuration format.

The interest of the method is to have 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 configuration file must be transferred manually in the target device, so it is less flexible than ‘Online’ mode.

**Figure 71. Offline beacon**

![Offline beacon diagram](image1)

**Beacon over-the-air configuration**

This method is similar to Offline mode, but the tool uses the OTA procedure to transfer the configuration file directly into the target Flash. A 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**

![Beacon over-the-air configuration diagram](image2)

**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.
3.6.3 Configuration of 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 configuration method.

Common parameters

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

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

In order to save batteries, the power level of 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 sufficient to estimate the real distance:

- Some beacon may transmit with high power, while other are using low power.
- The design of beacon antenna may be more or less 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 ‘public address’. The address entered is used.
- Tick the ‘random address’ check box. A random address is used.
- If nothing is selected, the board default public address is used.

**iBeacon parameters**

![iBeacon parameters]

First, select the type: iBeacon (default choice)

The user must check the Apple web site 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 BLE SIG group assigned values. For iBeacon, the Apple value is used: 0x004C. The assigned values are available on the SIG web site: https://www.bluetooth.com/specifications/assigned-numbers/company-identifiers


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 (see Transfer the configuration).
Eddystone UID parameters

Figure 76. Eddystone UID

<table>
<thead>
<tr>
<th>Select Type</th>
<th>Eddystone UID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beacon Name Space</td>
<td>4f31fd7f99000a0c494</td>
</tr>
<tr>
<td>Beacon Instance</td>
<td>000000000007</td>
</tr>
<tr>
<td>Enable TLM</td>
<td></td>
</tr>
</tbody>
</table>

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

- **NameSpace**, 10 bytes. Used to group some beacon in a logical pool. The way to generate the value is described by Google, see 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 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 in not known by the tool. So it must be managed directly by the firmware. As a consequence:

- The TLM option is not used for Online configuration mode.
- For Off Line and OTA mode, a bit is set in the configuration file. (see annex A)

Eddystone URL parameters

Figure 77. Eddystone URL

<table>
<thead>
<tr>
<th>Select Type</th>
<th>Eddystone URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>URL Prefix</td>
<td><a href="http://www">http://www</a>.</td>
</tr>
<tr>
<td>URL</td>
<td>st.com</td>
</tr>
<tr>
<td>Enable TLM</td>
<td></td>
</tr>
</tbody>
</table>

The Eddystone URL format is just sending an URL in the advertising message. In order to optimize space, start and end of 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 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 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.

Figure 81. OTA mode transfer configuration select device

<table>
<thead>
<tr>
<th>Configuration mode</th>
<th>OTA</th>
<th>iBeacon</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device configuration</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **0x0000FAE20025 - BEACON Demo - OTA enabled**
- **Beacon parameter address**: 0x6000

![Configuration mode - OTA - iBeacon](image)

- **c) 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 OTA loader by the user’s action. If 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

<table>
<thead>
<tr>
<th>Configuration mode</th>
<th>OTA</th>
<th>iBeacon</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device configuration</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **0x0000FAE20025 - BEACON Demo - OTA enabled**
- **Beacon parameter address**: 0x3000

![Configuration mode - OTA - iBeacon](image)

- **After transfer, application is restarted, and 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.

![ACI Utilities panel](image)

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

![Select checkbox](image)

#### 3.7.1 Remote services discovering

The remote services discovery performs a scan of the remote devices in the area.
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 “SCAN” button to start the discovery

The search procedure starts, and it is possible to stop it using “STOP” button.
If no remote device is found, the tool indicates "No device found". Otherwise, the user chooses one of the device found in the "Select Device" box.

At this stage, the user performs another scan procedure upon request.
- Click on the back button

Or connect to the selected remote device, by clicking on "CONNECT" button.
Figure 90. Connecting

If connection fails, an error is displayed.

Figure 91. Connection error

Once connected, the connect icon appear 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.
The user can select a characteristic and, depending on each of them, can read or write a value and be notified of a value changed. Note that read and write long characteristic are not supported neither authenticated signed write.

To read a value, the user clicks on "Read button".

To write a value, user enters the new value and click on "WRITE" button.
There are two ways to be informed on a value change, either via indicate method or via notification depending on the method property supported by the remote device.

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

To receive notification upon value change, the user can click on "NOTIFY" button.
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 the same time. To stop notification (resp. indication), the user can click on "UN-NOTIFY button (resp. "UN-INDICATE").

On disconnection, all registered notifications are removed.
3.7.2 Advertising

Figure 100. Advertising parameters

To activate 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 advertising interval
7. Enter optional slave connection interval
8. Click on "START ADVERTISING" button to start the procedure

The search procedure starts, advertising icon appear in blue, it is possible to stop it using "STOP ADVERTISING" button.
The connect icon may appear in blue if a remote device connects. In that case, advertising stops.
4 OpenThread mode

4.1 Presentation

4.1.1 Panel

The OpenThread main panel is organized with two tabs, ‘Commands’ and ‘Scripts’.

Figure 103. OpenThread - Command tab
Those 2 tabs have one common bottom area, the terminal area.

The terminal area is used to show the messages exchanged between application and target. We can see the commands sent to target and the responses received from 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 press enter to send command. The command sent with this line are recorded in 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 parameters area to fill the line, then the user can modify the line and send command with enter key. The commands list and parameters area are described in 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.
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.

The buttons ‘Start script recording’ and ‘Add pause in script’ allow saving a script. This part is described in Section 4.3: OpenThread scripts tab.
The command list is arranged in alphabetical order, and accessible from tree, for example below coap command we find coap resource, coap start and coap stop commands.

The command name and the definition are in the upper part of the command details area. Below 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 for launching script stored in text file.
The scripts use the same syntax as BLE scripts. The OpenThread specificities are described in this chapter. Please consult the BLE script description in Section 3.4: Scripts on page 39 for general information.
4.3.1 OpenThread script example

Figure 111. Sample script

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

#Pause commande
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 BLE, 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.
5 802.15.4 RF test mode

5.1 Presentation

The RF tests 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 set the device in reception.
- The PER test set the device in reception and one additional device is used as packet generator.

![Test mode selection](image)

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

To change the mode, it is necessary to come back to this panel. There are one "Back" key and one breadcrumb link in each test panel to come back to this panel "Test mode selection".
5.2 Transmitter (TX) test mode

This test mode configures the 802.15.4 device in emission. This emission is continuous (without packet) at one power level and one frequency.

Figure 113. Transmitter test mode

The user must:

- Select the power level (+6 dBm to -21 dBm).
- Select the TX frequency (channel11 - 2405MHz to channel 26 - 2480 MHz).
- Press "Start TX" key, the emission starts, press "Stop TX" key to interrupt the test.

5.3 Receiver test (RX) mode

This test mode configures the device in reception.
The user must:
- Select one frequency in the pick list
- Press Start RX key: the device is configured in the receiver mode at the selected frequency.
- Press Stop RX key: the device stops the reception and supplies the result, RSSI, LQI and number of frames. Those results appear in the low side.

5.4 Packet error rate (PER) mode

This mode configures the device in reception and one other device to play the role of 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 number of frames sent (Number of frames received/Number of Frames sent)*100
Four steps are necessary:

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

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

![Packet tester connection](image)

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

When second device is connected, it is not possible to change mode. The user need 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 tester.

Figure 116. 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". Only the single measurement mode uses this parameter, no continuous or multiple channels modes. It is applied to tester device.

Click on 'CONFIGURE DUT' to set the Device Under Test configuration.
5.4.3 Configure the parameters of device under test (DUT).

Figure 117. 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 118. PER test parameters

Three measurement modes are available:

- **Single measurement** measures once the frames number defined. The frequency of the tester is the one defined in panel "PER tester configuration" (TX frequency). The frequencies of DUT is as defined in panel "DUT configuration".

- **Continuous measurement** repeats the measurement on frames number until the user press "Stop Test" key. The frequency of DUT and Tester is the same, it's the one defined in the panel DUT configuration.

- **Multiple channels** measures the frequency defined in "Fill channel list" field. The default values are "11-26" this means all channels in 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 "Stop Test" key.

The result of continuous and multiple channels measurements can be saved in a csv file. The user must check the "Save test verdict in file" check box, and must define the name of file by "SELECT FILE" key before starting the test.
Table 4. Measurement setting

<table>
<thead>
<tr>
<th>Measurements</th>
<th>Single</th>
<th>Continuous</th>
<th>Multiple channels</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continuous measurement check box</td>
<td>Unchecked</td>
<td>Checked</td>
<td>Checked</td>
</tr>
<tr>
<td>Multiple Channels check box</td>
<td>Unchecked</td>
<td>Unchecked</td>
<td>Checked</td>
</tr>
<tr>
<td>Save test verdict in file check box</td>
<td>Not available</td>
<td>Available</td>
<td>Available</td>
</tr>
</tbody>
</table>

Three display modes are available:

1. Standard display

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

![Figure 119. Standard display](image1)

2. Chart display

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

![Figure 120. Chart display](image2)
3. Large display

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

**Figure 121. Large PER display**

![Large PER display]

**Figure 122. Large RSSI display**

![Large RSSI display]

**Figure 123. Large LQI display**

![Large LQI display]
Appendix A  Beacon configuration format

The beacon configuration file is a binary file. Its content is explained in Table 5.

<table>
<thead>
<tr>
<th>Byte #</th>
<th>Name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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 address type=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>
<thead>
<tr>
<th>Date</th>
<th>Revision</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>27-Nov-2017</td>
<td>1</td>
<td>Initial version</td>
</tr>
</tbody>
</table>
| 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 3.5.3: Advertising change for OTA in ST example |
| 23-Aug-2018 | 3        | Complete content reorganized to explain tool support to the original BLE 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 |
IMPORTANT NOTICE – PLEASE READ CAREFULLY

STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information on ST products before placing orders. ST products are sold pursuant to ST’s terms and conditions of sale in place at the time of order acknowledgement.

Purchasers are solely responsible for the choice, selection, and use of ST products and ST assumes no liability for application assistance or the design of Purchasers’ products.

No license, express or implied, to any intellectual property right is granted by ST herein.

Resale of ST products with provisions different from the information set forth herein shall void any warranty granted by ST for such product.

ST and the ST logo are trademarks of ST. All other product or service names are the property of their respective owners.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.

© 2019 STMicroelectronics – All rights reserved