UM2058

User manual

BlueNRG GUI SW package

Introduction

This document describes the BlueNRG GUI SW package (STSW-BNRGUI) with the BlueNRG graphical user interface (GUI) and the standalone script launcher utility.

The BlueNRG GUI is a PC application that can be used to interact and evaluate the capabilities both of the BlueNRG and BlueNRG-MS Bluetooth low energy (BLE) network processors: low power Bluetooth® smart ICs, compliant with the Bluetooth® specifications, with support both for master and slave roles. It also supports the BlueNRG-1 and BlueNRG-2 Bluetooth low energy systems-on-chip: low power Bluetooth® smart ICs compliant with the Bluetooth® specifications, both in master and slave roles.

The BlueNRG GUI allows standard and vendor-specific HCI commands to be sent to the device controller and events to be received from it. It also provides a script engine, which can be used to load and run user scripts based on Bluetooth low energy stack APIs and related stack events. These scripts can also be run via standalone script launcher utility through a PC DOS window, outside the BlueNRG GUI context. A set of sample scripts is provided in the SW package.

1 Getting started

This section describes all system requirements to run the BlueNRG GUI or script launcher utility, as well as the relative SW package installation procedure.

1.1 System requirements

The BlueNRG GUI PC application has the following minimum requirements:

- PC with Intel® or AMD® processor with administration rights running one of the following Microsoft® operating systems:
  - Windows 7, 10
- At least 128 Mb of RAM
- 2 USB ports
- 40 Mb of free hard disk space
- Adobe Acrobat Reader 6.0 or later

1.2 BlueNRG GUI SW package setup

1. Extract the content of the en.STSW-BNRGUI.zip file into a temporary directory.
2. Extract and launch the BlueNRG_GUI_-x.x.x.Setup.exe file and follow the on-screen instructions.

1.3 BlueNRG GUI SW package structure

The BlueNRG GUI SW package files are organized into the following folders:

- **Application**: contains BlueNRG_GUI.exe and BlueNRG_Script_Launcher.exe PC applications and some sample scripts.
- **Docs**: contain the BlueNRG GUI SW package release notes and html documentation covering ACI, events and sample scripts.
- **Firmware**:
  - contains the DTM and USB to serial binary files to be used for the BlueNRG-1 and BlueNRG-2 development kit.
  - contains the BlueNRG_VCOM_x.x.hex prebuilt binary file to be used for the STM32L1 microcontroller on BlueNRG and BlueNRG-MS kit motherboards. It also contains the BlueNRG and BlueNRG-MS FW stack versions.
- **PCDriver**: contains the PC DFU and USB Virtual COM (Windows 7) drivers.

Note: The provided BlueNRG_VCOM_x.x and DTM binary files can be rebuilt using the Virtual_COM_Port and DTM projects, respectively, available in the STSW-BLUENRG-DK SW package (Table 8. Reference information). Each project provides a complete set of source and header files.

2 GUI software description

The BlueNRG GUI included in the software package is a graphical user interface that can be used to interact and evaluate the capabilities of the BlueNRG and BlueNRG-MS network processors, and the BlueNRG-1 and BlueNRG-2 systems-on-chip.

This utility can send standard and vendor-specific HCI commands to the controller and receive events from it. It allows the user to configure each field of the HCI command packets to be sent and analyzes all received packets to allow easy low level management of the BlueNRG, BlueNRG-MS, BlueNRG-1 and BlueNRG-2 devices.

2.1 Requirements

To use the BlueNRG GUI, ensure your hardware and software are set up (BlueNRG GUI installed) correctly.

2.1.1 BlueNRG and BlueNRG-MS network coprocessors

The STM32L1 in the STEVAL-IDB002V1,STEVAL-IDB005V1 and STEVAL-IDB005V1D kits has been preprogrammed with a demo application (BlueNRG sensor demo). Hence, new firmware must be loaded into the STM32L1 microcontroller in order to use the BlueNRG GUI. The firmware image that must be programmed is the latest BlueNRG_VCOM_x_x.hex available in the BlueNRG GUI SW package in Firmware/ STM32L1_prebuilt_images folder.

To download this binary image into the internal Flash of the STM32L1, the microcontroller must be set in the special DFU (device firmware upgrade) mode.

To enter DFU mode:

- BlueNRG and BlueNRG-MS development platforms (order codes: STEVAL-IDB002V1, STEVAL-IDB005V1, STEVAL-IDB005V1D):
  1. Power up the board
  2. Press and hold push button
  3. Reset the board using the RESET button (keep the push button pressed while resetting). The orange LED DL2 starts to blink
  4. Release the push button
  5. Use the BlueNRG GUI to flash the device with the BlueNRG_VCOM_x_x.hex binary file (Tools → Flash motherboard FW)
- BlueNRG and BlueNRG-MS USB dongles (order codes: STEVAL-IDB003V1, STEVAL-IDB006V1):
  1. press and hold the SW1 button
  2. plug the USB dongle on a PC USB port. The orange LED D3 starts to blink
  3. use BlueNRG GUI to flash the device with BlueNRG_VCOM_x_x.hex binary file (Tools → Flash Motherboard FW)

2.1.2 BlueNRG-1 and BlueNRG-2 network coprocessors

The STEVAL-IDB007Vx (x=1,2) (BlueNRG-1) and STEVAL-IDB008Vx (x=1,2) /STEVAL-IDB009Vx (x=1) (BlueNRG-2) kits are preprogrammed with a demo application (BLE sensor demo), so new firmware must be loaded into the BlueNRG-1 and BlueNRG-2 devices to configure them as network coprocessors and use the BlueNRG GUI.

The firmware images, that must be programmed in order to configure the devices on the STEVAL-IDB007Vx (x=1,2) (BlueNRG-1) and STEVAL-IDB008Vx (x=1,2) /STEVAL-IDB009Vx (x=1) (BlueNRG-2) boards as network coprocessors, are available in the BlueNRG GUI SW package in the *Firmware/BlueNRG1* and *Firmware/ BlueNRG-2* folders, respectively.

Two network coprocessor configurations are available:

1. UART (DTM binary file: DTM_UART.hex)
2. SPI (DTM binary file: DTM_SPI.hex)

To download the selected binary image into the STEVAL-IDB007Vx (x = 1,2) (BlueNRG-1) or STEVAL-IDB008Vx (x=1,2) /STEVAL-IDB009Vx (x=1) (BlueNRG-2) internal Flash:

1. Connect the device platform to a PC USB port

2. On PC computer window, use the the "drag and drop" mass storage upgrade capability for copying the selected binary image into the associated IDB007/8/9VX platform

Another way to download the selected binary image is the following:

1. Connect the device platform to a PC USB port
2. Open the BlueNRG Flasher tool available on STSW-BLUENRG1-DK SW package (Table 8. Reference information)
3. Select the COM port associated with the device platform to be configured as a network coprocessor
4. Select the specific DTM binary file through the select file button
5. Select the mass erase option
6. Press the Flash button to program the device

Notes:

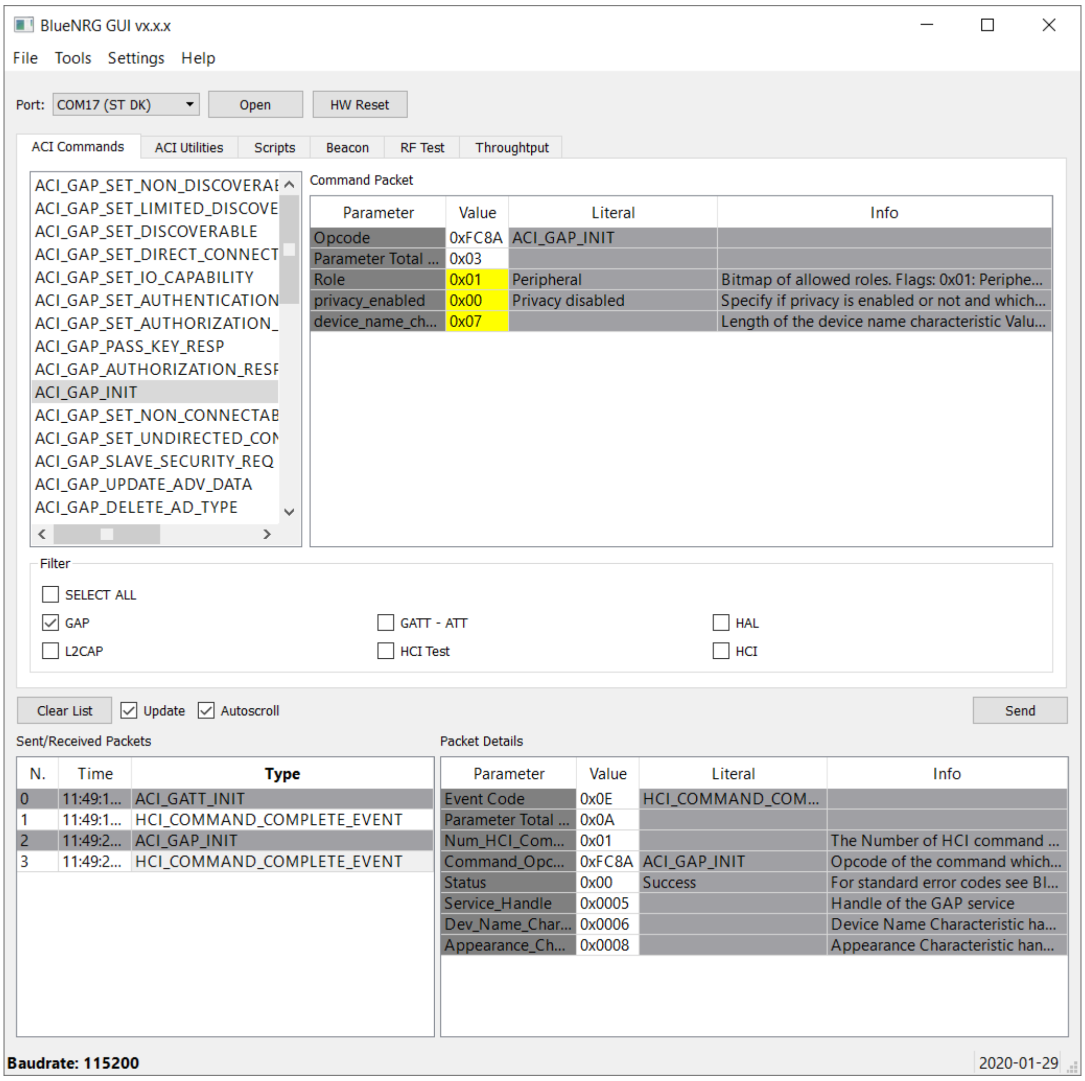
1. The BlueNRG GUI --> Tools --> Stack updater… utility is intended only to update a BlueNRG-1 or BlueNRG-2 device already configured as a network coprocessor as it assumes that specific updater/ bootloader code is already loaded on the device.
2. The DTM_UART.hex and DTM_SPI.hex binary image files include the updater code, which allows the update of the specific DTM FW version through the BlueNRG GUI --> Tools --> Stack updater… utility. DTM binary images without updater code are also provided in the SW package: DTM_UART_NOUPDATER.bin for UART configuration and DTM_SPI_NOUPDATER.bin for SPI configuration.
3. Once the STEVAL-IDB007Vx (x = 1,2) (BlueNRG-1) or STEVAL-IDB008Vx (x = 1,2) /STEVAL-IDB009Vx (x=1) (BlueNRG-2) platform has been configured as a network coprocessor, new DTM binary file versions can be updated directly using the BlueNRG GUI, Tools, Stack Updater … utility thus:
   a. Connect the device platform (STEVAL board) to a PC USB port
   b. Open the BlueNRG GUI tool
   c. Open COM port associated with the device platform
   d. Select BlueNRG, Tools, BlueNRG updater …
   e. Select the DTM binary images without updater available in Firmware/BlueNRG1 or Firmware/ BlueNRG-2 folders (DTM_UART_NOUPDATER.bin or DTM_SPI_NOUPDATER.bin)
   f. Select update to program the new DTM binary image
4. No other BlueNRG-1 or BlueNRG-2 application binary image can be programmed on a corresponding device using the BlueNRG GUI, Tools, Stack Updater utility.
5. In order to use the STEVAL-IDB007V1 for the BlueNRG-1 SPI network coprocessor configuration, please make sure the following hardware changes have been made on such platform: make a short at positions R59, R60, R61, R62 on STEVAL-IDB007V1 platform.
6. Document content is also valid for the BlueNRG-1, STEVAL-IDB007V1M evaluation platform based on the SPBTLE-1S module with 32 MHz HS crystal and for the BlueNRG-2, STEVAL-IDB008V1M kit (evaluation platform based on the BlueNRG-M2SA module with 32 MHz HS crystal). The associated DTM_UART.hex and DTM_SPI.hex binary images are provided within the folder Firmware/STEVAL-IDB007V1M.

2.2 The BlueNRG graphical user interface

This section describes the main functions of the BlueNRG GUI application. You can run this utility by clicking on the BlueNRG GUI icon on the desktop or under: start menu folder ST BlueNRG GUI X.X.X → BlueNRG GUI.

2.2.1 GUI main window

Figure 1. **BlueNRG GUI main window**



ACI_GAP_SET_NON_DISCOVERAE
ACI_GAP_SET_LIMITED_DISCOVE
ACI_GAP_SET_DISCOVERABLE
ACI_GAP_SET_DIRECT_CONNECT
ACI_GAP_SET_IO_CAPABILITY
ACI_GAP_SET_AUTHENTICATION
ACI_GAP_SET_AUTHORIZATION_
ACI_GAP_PASS_KEY_RESP
ACI_GAP_AUTHORIZATION_RESF
ACI_GAP_INIT
ACI_GAP_SET_NON_CONNECTAB
ACI_GAP_SET_UNDIRECTED_CON
ACI_GAP_SLAVE_SECURITY_REQ
ACI_GAP_UPDATE_ADV_DATA
ACI_GAP_DELETE_AD_TYPE

Command Packet

Parameter	Value	Literal	Info
Opcode	0xFC8A	ACI_GAP_INIT	
Parameter Total ...	0x03		
Role	0x01	Peripheral	Bitmap of allowed roles. Flags: 0x01: Periphe...
privacy_enabled	0x00	Privacy disabled	Specify if privacy is enabled or not and which...
device_name_ch...	0x07		Length of the device name characteristic Valu...

Sent/Received Packets

N.	Time	Type
0	11:49:1...	ACI_GATT_INIT
1	11:49:1...	HCI_COMMAND_COMPLETE_EVENT
2	11:49:2...	ACI_GAP_INIT
3	11:49:2...	HCI_COMMAND_COMPLETE_EVENT

Packet Details

Parameter	Value	Literal	Info
Event Code	0x0E	HCI_COMMAND_COM...	
Parameter Total ...	0x0A		
Num_HCI_Com...	0x01		The Number of HCI command ...
Command_Opc...	0xFC8A	ACI_GAP_INIT	Opcode of the command which...
Status	0x00	Success	For standard error codes see Bl...
Service_Handle	0x0005		Handle of the GAP service
Dev_Name_Char...	0x0006		Device Name Characteristic ha...
Appearance_Ch...	0x0008		Appearance Characteristic han...

Baudrate: 115200 2020-01-29

The BlueNRG GUI main window has different zones, some of which can be resized.

Port and interface selection

The uppermost zone allows the user to open the COM port associated to the BLE controller.

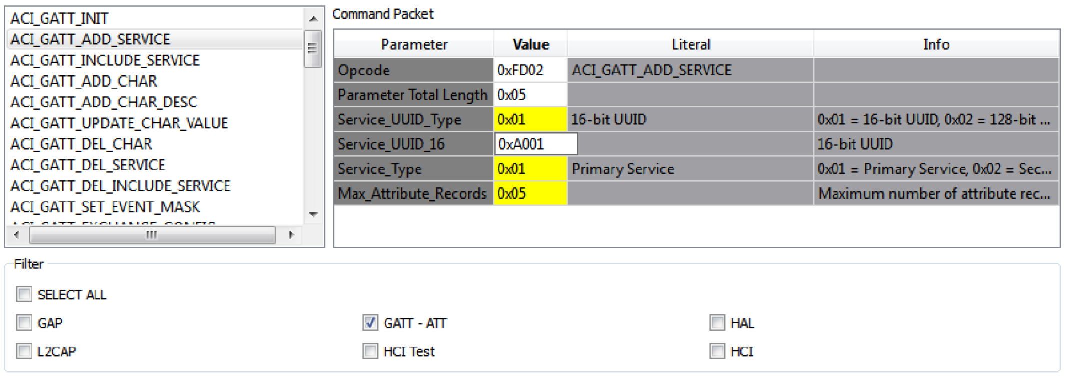
When a COM port is opened the following information is displayed:

- BlueNRG, BlueNRG-MS, BlueNRG-1 or BlueNRG-2 HW version
- BlueNRG or BlueNRG-MS FW stack version, or BlueNRG-1 or BlueNRG-2 FW stack and DTM FW versions
- Motherboard GUI firmware (BlueNRG_VCOM_x_x or USB to serial) version

HCI commands

The HCI commands tab contains a list of all the available HCI commands. Commands can be filtered by checking/unchecking boxes under the filter section. After clicking on one of the commands, all packet fields are displayed on the command packet table in the upper-right section of the tab.

Figure 2. **Command packet table**



ACI_GATT_INIT
ACI_GATT_ADD_SERVICE
ACI_GATT_INCLUDE_SERVICE
ACI_GATT_ADD_CHAR
ACI_GATT_ADD_CHAR_DESC
ACI_GATT_UPDATE_CHAR_VALUE
ACI_GATT_DEL_CHAR
ACI_GATT_DEL_SERVICE
ACI_GATT_DEL_INCLUDE_SERVICE
ACI_GATT_SET_EVENT_MASK

Command Packet

Parameter	Value	Literal	Info
Opcode	0xFD02	ACI_GATT_ADD_SERVICE	
Parameter Total Length	0x05		
Service_UUID_Type	0x01	16-bit UUID	0x01 = 16-bit UUID, 0x02 = 128-bit ...
Service_UUID_16	0xA001		16-bit UUID
Service_Type	0x01	Primary Service	0x01 = Primary Service, 0x02 = Sec...
Max_Attribute_Records	0x05		Maximum number of attribute rec...

Filter

- [ ] SELECT ALL
- [ ] GAP
- [x] GATT - ATT
- [ ] HAL
- [ ] L2CAP
- [ ] HCI Test
- [ ] HCI

The command packet table contains four columns:

- **Parameter**: name of the packet field as per volume 2, part E of Bluetooth specification
- **Value**: field value represented in hexadecimal format (right-click on a cell to change the format)
- **Literal**: meaning of the current field value
- **Info**: description of the corresponding field

Only the yellow cells of this table can be modified by the user. *Parameter Total Length* is fixed or automatically calculated after modifying cell content.

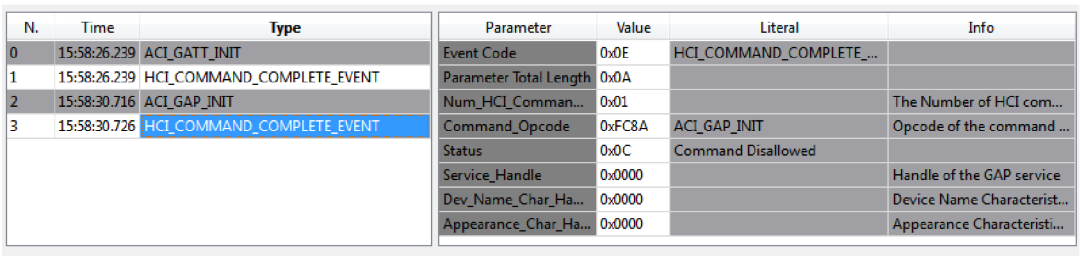
After fields have been modified, the command can be sent using the send button.

HCI packet history and details

Two frames at the bottom of the main window show packets sent to and received from the BLE controller, as well as other events:

- The frame on the left (sent/received packets) holds a history of all packets
- The frame on the right (packet details) shows all the details of the selected packet as per the command packet table

Figure 3. **Packet history and details**

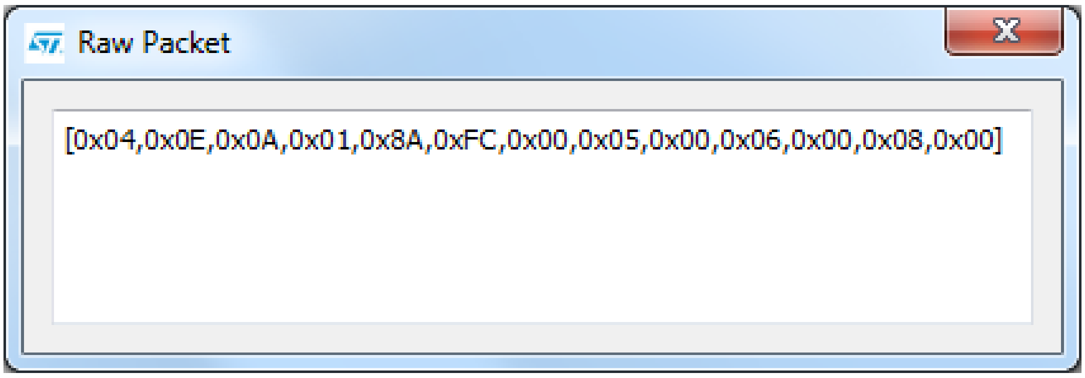


N.	Time	Type
0	15:58:26.239	ACI_GATT_INIT
1	15:58:26.239	HCI_COMMAND_COMPLETE_EVENT
2	15:58:30.716	ACI_GAP_INIT
3	15:58:30.726	HCI_COMMAND_COMPLETE_EVENT

Parameter	Value	Literal	Info
Event Code	0x0E	HCI_COMMAND_COMPLETE_...	
Parameter Total Length	0x0A		
Num_HCI_Comman...	0x01		The Number of HCI com...
Command_Opcode	0xFC8A	ACI_GAP_INIT	Opcode of the command ...
Status	0x0C	Command Disallowed	
Service_Handle	0x0000		Handle of the GAP service
Dev_Name_Char_Ha...	0x0000		Device Name Characterist...
Appearance_Char_Ha...	0x0000		Appearance Characteristi...

Double-clicking on a row of the sent/received packet table shows the raw packet.

Figure 4. **Raw packet dump**



```
[0x04,0x0E,0x0A,0x01,0x8A,0xFC,0x00,0x05,0x00,0x06,0x00,0x08,0x00]
```

Some events (displayed in yellow cells) can provide other information. HCI packets sent towards the BLE controller are displayed in gray cells while received packets are shown inside white cells.

The sent/received packets table can be cleared by clicking on clear list button. Update and auto-scrolling check boxes enable or disable updating and auto-scrolling of the sent/received packets table while new packets are sent or received (however, information is still printed).

The sent/ received packets can be stored and later reloaded on the GUI, by using the utilities provided on file menu:

- **Save History**: saves the current list of sent commands and received events to a CSV file or to a simple text file
- **Load History**: loads a list of sent commands and received events from a CSV file
- **Save as Python Script**: stores the current list of sent commands and received events as a script file (python format), which can be (by adding specific code for handling events, parameters) and then used in the GUI script window as part of a user application scenario (refer to Section 2.2.4 GUI script window)

- **Save As C Code...**: stores the current list of APIs and events on sent/received packet table as header and source files (C language), which can be used as starting point for user C code development. The generated header and source files are provided within a complete toolchains framework (IAR, KEIL Atollic), which allows user to get a basic reference code to be customized and integrated for addressing the specific user application scenario (STM32L4, BlueNRG-1,2 network coprocessor framework is also generated). User is requested to:
  1. On the BlueNRG-1, BlueNRG-2 devices, add the configuration header file with proper values required for the correct BlueNRG-1,2 BLE radio initialization, based on specific user application scenario:
     a. For BLE stack v2.0, v2.1 or further, this file can be generated using the BlueNRG-X Radio Initialization Parametes Wizard available on the BlueNRG-1_2 SW package (STSW-BLUENRG1-DK).
     b. Save As C Code ... window also allows user to directly launch the BlueNRG-X Radio Initialization Parametes Wizard in order to define the radio initialization parameters and store the associated user configuration file on the generated inc folder.
  2. On the BlueNRG-1, BlueNRG-2 devices, aci_gap_init() API, make sure that the device_name_char_len parameter is configured exactly with the actual device name characteristic length
  3. On the BlueNRG-1, BlueNRG-2 devices, make sure that each aci_gatt_add_service(), Max_Attribute_Records parameter is configured exactly with the required number of attributes for the user application scenario, in order to avoid API failure:
     a. For each defined service, the correct Max_Attribute_Records parameter value is reported on the BlueNRG-X radio Init wizard, output window #define MAX_NUMBER_ATTRIBUTE_SERVICEx (x = 1, .. number of added services): just use the associated define numeric value
  4. On the BlueNRG-1, BlueNRG-2 devices, BLE stack v2.1 the generated IDE projects are built with the full stack modular configuration option (BLE_STACK_CONFIGURATION=BLE_STACK_FULL_CONFIGURATION). User can customize it according to his specific application scenario.
  5. Customize APP_Tick() function with user state machine and associated states
  6. Customize the required event callback functions taking into account application states
  7. Remove possible duplications of event callback functions
  8. Customize APIs execution sequence based on application state machine, event callbacks and state updates (i.e.: start service or characteristic discovery, read, write, notification,.. once the device is connected: connection complete event callback has been called)
  9. Remove repeated calls to the same APIs, if not needed (or review code by adding a simple for/while code iteration)
  10. Customize API input parameters with proper references to application variables
  11. Customize interrupt service routines based on user application needs (refer to generated default user functions section)
  12. Customize, modify generated default user functions
  13. Replace the linker files with the latest available version on associated DK SW packages
  14. Enable debug messages by setting DEBUG 1 on generated source files

After the BLE_User_main.c, user.c, user.h has been generated, look for the comments with prefix USER_ACTION_IS_NEEDED. These comments highlight some possible post-built code customization to be performed on user side, in order to address his specific application scenario.

2.2.2 Tools

The BlueNRG GUI has some functions that can be accessed through the tools menu. These tools are described in this section.

Stack updater

This tool can be used to update the firmware in the BlueNRG and BlueNRG-MS devices by using their internal bootloader. The BlueNRG_VCOM_x_x.hex firmware must be present on the BlueNRG or BlueNRG-MS motherboard STM32L1 microcontroller and the COM port must be open.

To use this tool:

1. Go to tools → Stack updater
2. Select the correct stack firmware (.img) for BlueNRG or BlueNRG-MS device
3. Press update to start the update procedure

If the procedure completes with no errors, the new firmware has been loaded into the device internal Flash.

This tool can also be used to update the network coprocessor firmware inside the BlueNRG-1 or BlueNRG-2 device, provided that it has been already configured as a network coprocessor with a DTM FW image containing the update functionality (refer to Section 2.1.2 BlueNRG-1 and BlueNRG-2 network coprocessors).

IFR

To preserve the flexibility of the BlueNRG or BlueNRG-MS device, the firmware uses a parameter configuration table in a sector of the Flash called information register (IFR). The BlueNRG GUI IFR tool can read and modify this portion of BlueNRG, BlueNRG-MS Flash.

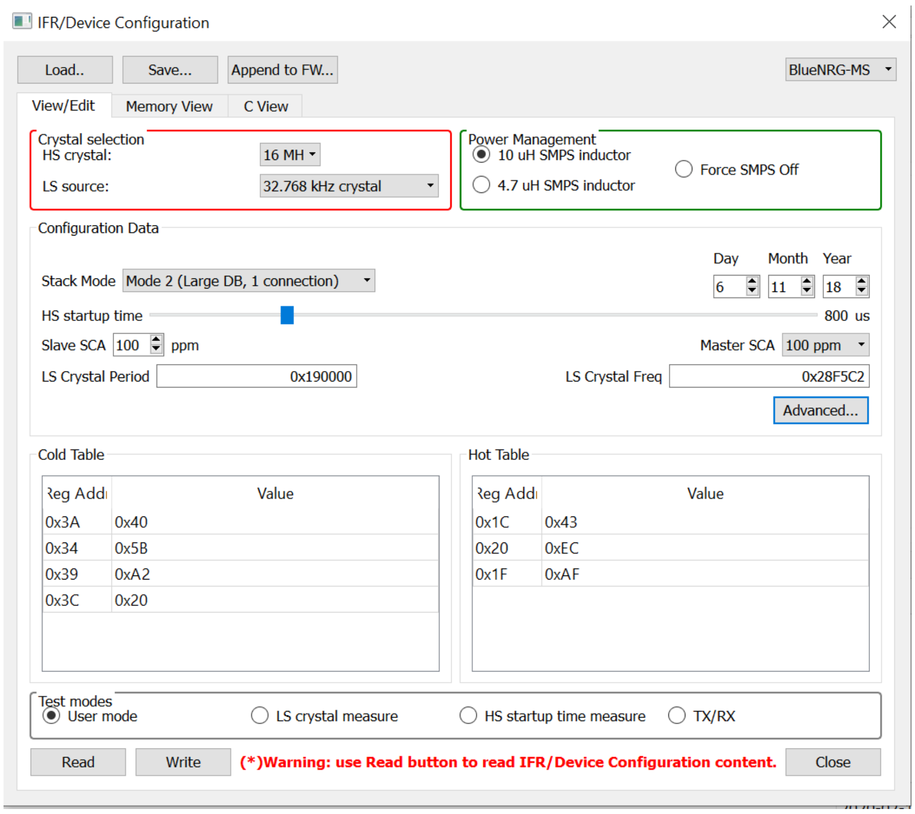
This tool is available in BlueNRG GUI → Tools → IFR/device configuration → item.

The BlueNRG GUI IFR utility helps you define the IFR data and represents the only supported mode to define IFR data based on customer needs.

The utility provides the following windows:

- **View/Edit view**: displays the IFR regions with related fields and description. The user can modify some of these fields according to his needs.
- **Memory view**: displays the IFR field memory addresses and related values that are generated by BlueNRG GUI according to the specified values.
- **C view**: displays the C language structure related to the IFR configuration data region matching the View/Edit and Memory view.

Figure 5. **BlueNRG GUI IFR/device configuration tool: view/edit view for the BlueNRG, BlueNRG-MS devices**



BlueNRG-MS

Crystal selection
HS crystal: 16 MH
LS source: 32.768 kHz crystal

Power Management
[✓ 10 uH SMPS inductor] 4.7 uH SMPS inductor Force SMPS Off

Configuration Data
Stack Mode: Mode 2 (Large DB, 1 connection)
Day 6 Month 11 Year 18
HS startup time: 800 us
Slave SCA: 100 ppm
Master SCA: 100 ppm
LS Crystal Period: 0x190000
LS Crystal Freq: 0x28F5C2

Cold Table

Reg Addr	Value
0x3A	0x40
0x34	0x5B
0x39	0xA2
0x3C	0x20

Hot Table

Reg Addr	Value
0x1C	0x43
0x20	0xEC
0x1F	0xAF

Test modes
[✓ User mode] LS crystal measure HS startup time measure TX/RX

(*)Warning: use Read button to read IFR/Device Configuration content.

In the view/edit view, the following operations are available:

- Select high speed (HS) crystal (16 or 32 MHz) and low speed oscillator source (32 kHz or the internal ring oscillator)
- Set power management options (SMPS inductor or SMPS off configuration)
- Change stack mode. Each mode has a different functionality:
  - Mode 1: slave/master, 1 connection only, small GATT database (RAM2 off during sleep)
  - Mode 2: slave/master, 1 connection only, large GATT database (RAM2 on during sleep)
  - Mode 3: slave/master, 8 connections, small GATT database (RAM2 on during sleep)
  - Mode 4: slave/master, simultaneous advertising and scanning, up to 4 connections, small GATT database (RAM2 on during sleep) (only on BlueNRG-MS FW stack versions above 7.1a)
- Change HS start-up time parameter. This parameter controls the time offset between the wakeup of the device and the start of RX/TX phase. It must be big enough to allow the device to be ready to transmit or receive after wakeup from sleep. This time depends on the start-up time of the high speed crystal
- Change sleep clock accuracy. This must reflect the actual clock accuracy, depending on the low speed oscillator or crystal in use
- Set low speed (LS) crystal period and frequency
- View/change date to distinguish between different versions of configurations
- View registers that are written into the radio (hot and cold table)
- Set some test modes for specific tests
- Read IFR content from the BlueNRG and BlueNRG-MS devices
- Write IFR configuration to the BlueNRG and BlueNRG-MS devices

The following general utilities are also available:

- Load button: loads a configuration file
- Save button: saves the current parameters into a configuration file

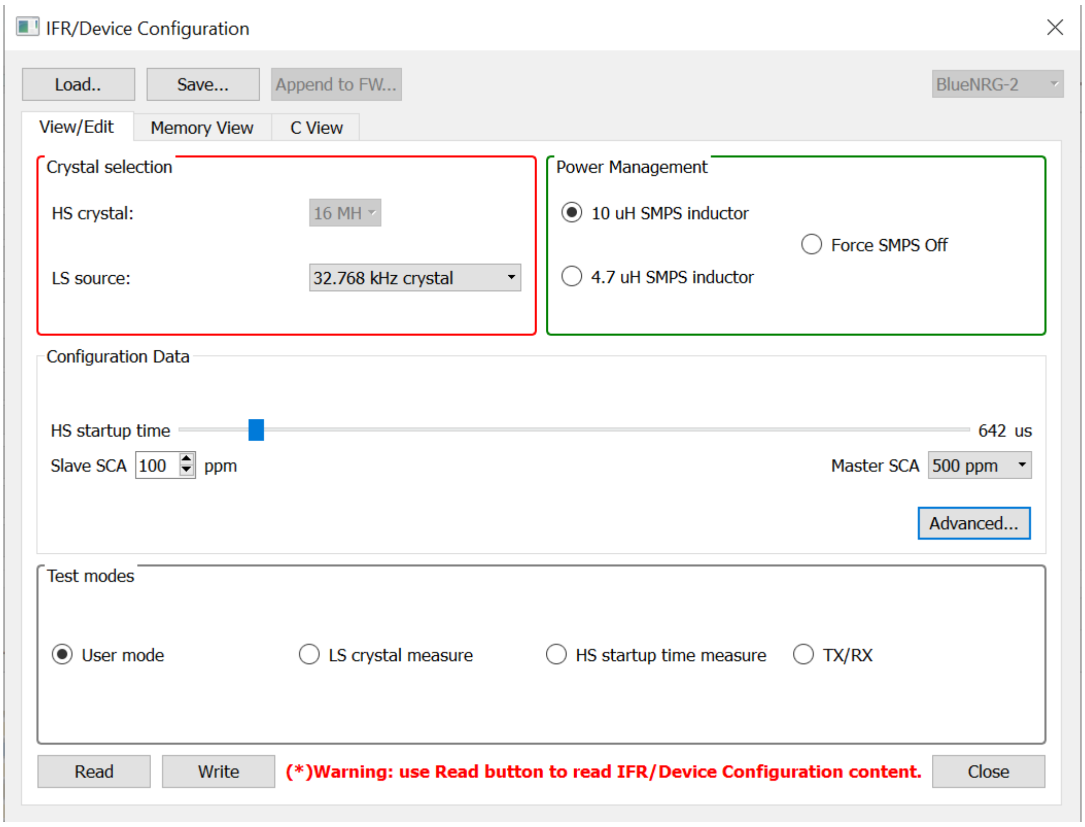
Device configuration for the BlueNRG-1 and BlueNRG-2

To preserve the flexibility, the BlueNRG-1 and BlueNRG-2 device firmware uses a hardware configuration table, which can be read and modified by BlueNRG GUI IFR/device configuration tool; this tool is available in BlueNRG GUI, Tools, IFR/device configuration item.

The utility provides the following windows:

- **View/Edit view**: displays the device configuration parameter regions with related fields and description. The user can modify some of these fields as required
- **Memory view**: displays the device configuration field memory addresses and related values that are generated by BlueNRG GUI according to the specified values
- **C view**: displays the C language structure related to the device configuration data region matching the View/ Edit and Memory view

Figure 6. **Device configuration**


IFR/Device Configuration

Load.. | Save... | Append to FW... | BlueNRG-2

View/Edit | Memory View | C View

Crystal selection
HS crystal: 16 MH
LS source: 32.768 kHz crystal

Power Management
[✓ 10 uH SMPS inductor]
Force SMPS Off
4.7 uH SMPS inductor

Configuration Data
HS startup time 642 us
Slave SCA 100 ppm
Master SCA 500 ppm
Advanced...

Test modes
[✓ User mode] LS crystal measure HS startup time measure TX/RX

Read | Write | (*)Warning: use Read button to read IFR/Device Configuration content. | Close

In the View/Edit view, only the following operations are supported:

1. Select low speed oscillator source (32 kHz or the internal ring oscillator)
2. Set power management options (SMPS inductor value or SMPS off configuration)
3. Set some test modes for specific tests
4. Read device configuration parameter content from the BlueNRG-1 or BlueNRG-2
5. Write device configuration parameter configuration to the BlueNRG-1 or BlueNRG-2

Note: *The BlueNRG-1 or BlueNRG-2 DTM UART HS crystal (16 or 32 MHz) is selected at compile time and cannot be changed through BlueNRG GUI IFR/device configuration.*

The following general utilities are also available:

- Load button: allows loading a configuration file
- Save button: allows saving the current parameters into a configuration file

Flash motherboard firmware

The BlueNRG GUI embeds a utility that let you Flash firmware to the STM32L1 microcontroller on the BlueNRG or BlueNRG-MS motherboard without a JTAG/SWD programmer. This utility uses a bootloader (DFU) located in the first 12 kB of the STM32L1 Flash.

Any application to be programmed to the STM32L1 by this tool must first consider that the lower area of the Flash is used by the bootloader (two precautions must be taken for any STM32L1 firmware: 1) change memory regions in linker script (vector table and Flash must start at 0x08003000); 2) change the vector table offset (NVIC_SetVectorTable()))

On the BlueNRG-1 and BlueNRG-2 development platform kits, this utility let you upgrade the USB-to-Serial firmware when needed.

To do so, activate the DFU application manually:

1. Press and hold the RESET button
2. Plug the USB cable to the board
3. Release the RESET button
4. Red LED DL2 blinks to confirm the DFU application is running
5. On BlueNRG GUI PC application, select Tools → Flash Motherboard FW...
6. Press the apply button of the window
7. Select the firmware USB_to_SERIAL.hex available in the firmware/BlueNRG-1 or firmware/BlueNRG-2 folder and press the open button
8. Wait for the end of the upgrade operation

OTA bootloader

The OTA bootloader let you Flash new firmware to the STM32L1 on a remote BlueNRG or BlueNRG-MS motherboard via Bluetooth low energy technology. It also let you Flash new firmware to a remote BlueNRG-1 or BlueNRG-2 device via Bluetooth low energy technology (refer to the dedicated device application notes available on www.st.com for further information).

Get production data

From the tools menu it is possible to retrieve production information stored at platform manufacturing time. This data is stored in the EEPROM available on the platform kits.

Get version

The Get version tool is used to retrieve the version of the BlueNRG GUI firmware (BlueNRG_VCOM_x_x.hex) on the STM32L1 controller, USB to serial FW version on the BlueNRG-1 and BlueNRG-2 platform, and hardware and firmware versions from the BlueNRG, BlueNRG-MS, BlueNRG-1 and BlueNRG-2 devices.

Settings

This tool let you configure the firmware stack version to be used from the GUI (when no device is actually connected to a PC USB port). It also let you configure the GUI serial baud rate (serial UART communication only).

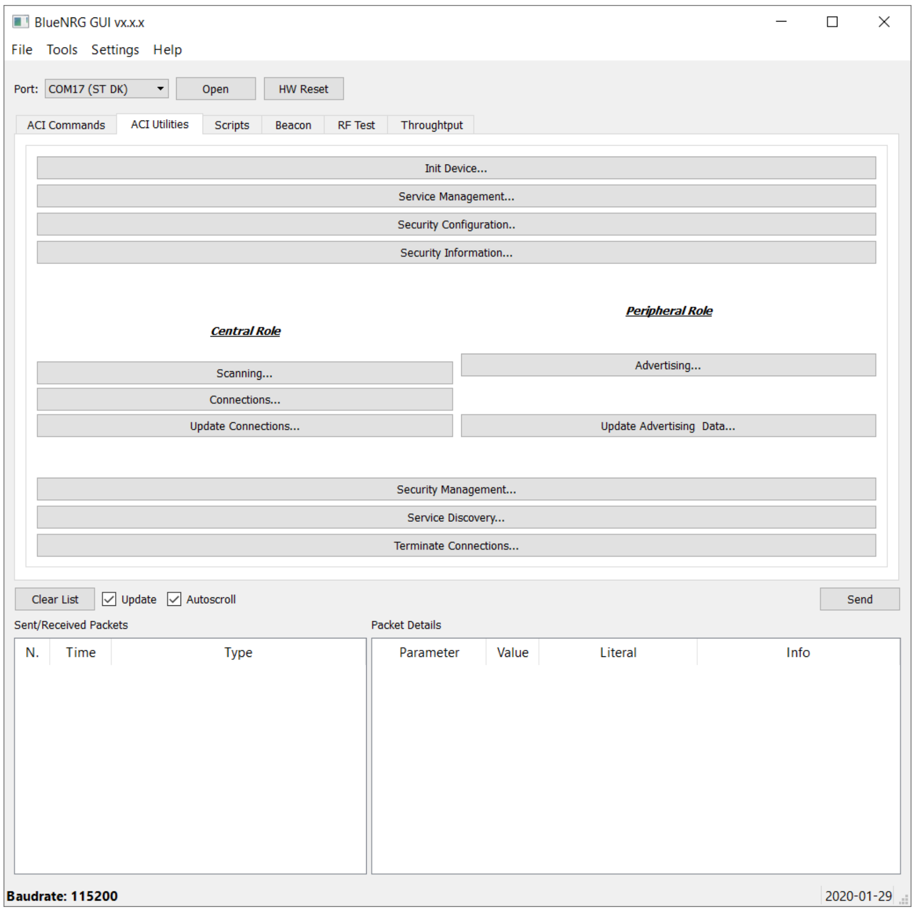
In order to use this function.

1. Go to settings → select FW 7.x or 7.1 for the BlueNRG-MS device, FW 6.4 for the BlueNRG device, FW 1.0 or FW 2.x for the BlueNRG-1 device and 2.x for the BlueNRG-2 device
2. Go to settings → select Set Baud Rate… and choose the value

2.2.3 GUI ACI utility window

The BlueNRG GUI ACI utility window provides several tabs to allow testing of some application scenarios.

Figure 7. BlueNRG GUI ACI utility window



Central and peripheral roles are supported with the BLE operations described in the following tables.

Table 1. **GUI ACI utilities window: available general operations**

Operation	Associated actions	Notes
Init Device…	Allows a device to be initialized by selecting: - Role - Stack mode (1,2,3,4) - Address type (public, random) and value - Tx power level - Power mode - Device name	
Security configuration	Allows the device security database to be erased Allows security settings to be configured: - IO capabilities - Authentication requirements	
Service management…	Allows adding a service by selecting: - UUID type (16 or 128 bits) - Service type (primary or secondary) - Set max. number of records For each service, it allows a characteristic and related descriptors to be added by selecting: - UUID type (16 or 128 bits) - Properties - Security permissions - Variable length or not - Length - GATT event mask - Encryption key size	After a characteristic is defined, the user can edit its parameters and/or delete it. Once a service and its characteristics, descriptors have been defined, click OK to add them to the GATT database. The defined GATT database is shown on a specific view. The "Save View" button allows the defined services and associated characteristics to be stored in order to be reloaded later with the "Load View" button
Security management ...	Allows security operations (send slave security request, send pairing request, allow rebond, ...) to be performed	
Security information	Allows the list of the bonded devices to be got	
Service discovery ..	Allows discovery of all services and related characteristics of available connections.	Service start handle, end handle and UUID are showed. For each selected service the related characteristics information are showed (attribute handle, property, value handle and UUID). For the available characteristic with notify or indication property, it is possible to enable the notification/indication. It is also possible to store as view the discovered services and associated characteristics for a later usage through the "Load View" button on "Service Management …" window. This allows user to build his own "Services/ Characteristics View" database from discovered peer device services and characteristics, and store them for possible future application scenarios.
Terminate connection...	Allows terminating the available connections	

The "Service Management..." window provides the load and save view buttons. A set of predefined views related to standard services and profiles are available on the BlueNRG GUI SW package, Application/Services_View folder ("Alert Notification Server, Blood Pressure Sensor, Glucose Sensor, Heart Rate Sensor, Health Thermometer Sensor, Find Me Target, Phone Alert Server, HID, Time, …")

User can load them through the "Load View" button and add to his application database.

Table 2. **GUI ACI utility window: available central operations**

Operation	Associated actions	Notes
Scanning	Allows placing the device in scanning mode by selecting: - GAP procedure (limited, general, general-connection establishment and terminate general-connection establishment procedures) - Enable or disable filters - Set own address type - Set passive or active scan - Set scanning interval and window	
Connections	Allows connecting to a peer device by: - Searching for the devices in advertising - Selecting the device to which to connect - Selecting the connection parameters - Peer address and type - Scan interval and window - Connection interval (min. and max.) - Latency - Supervision timeout - Connection event length (min. and max.)	The addresses of the detected advertising devices are displayed
Update connections	Allows updating the connection parameters of available connections by: - Selecting the specific connection to be updated - Setting the new connection parameters - Connection interval (min. and max.) - Latency - Supervision timeout - Connection event length (min. and max.)	

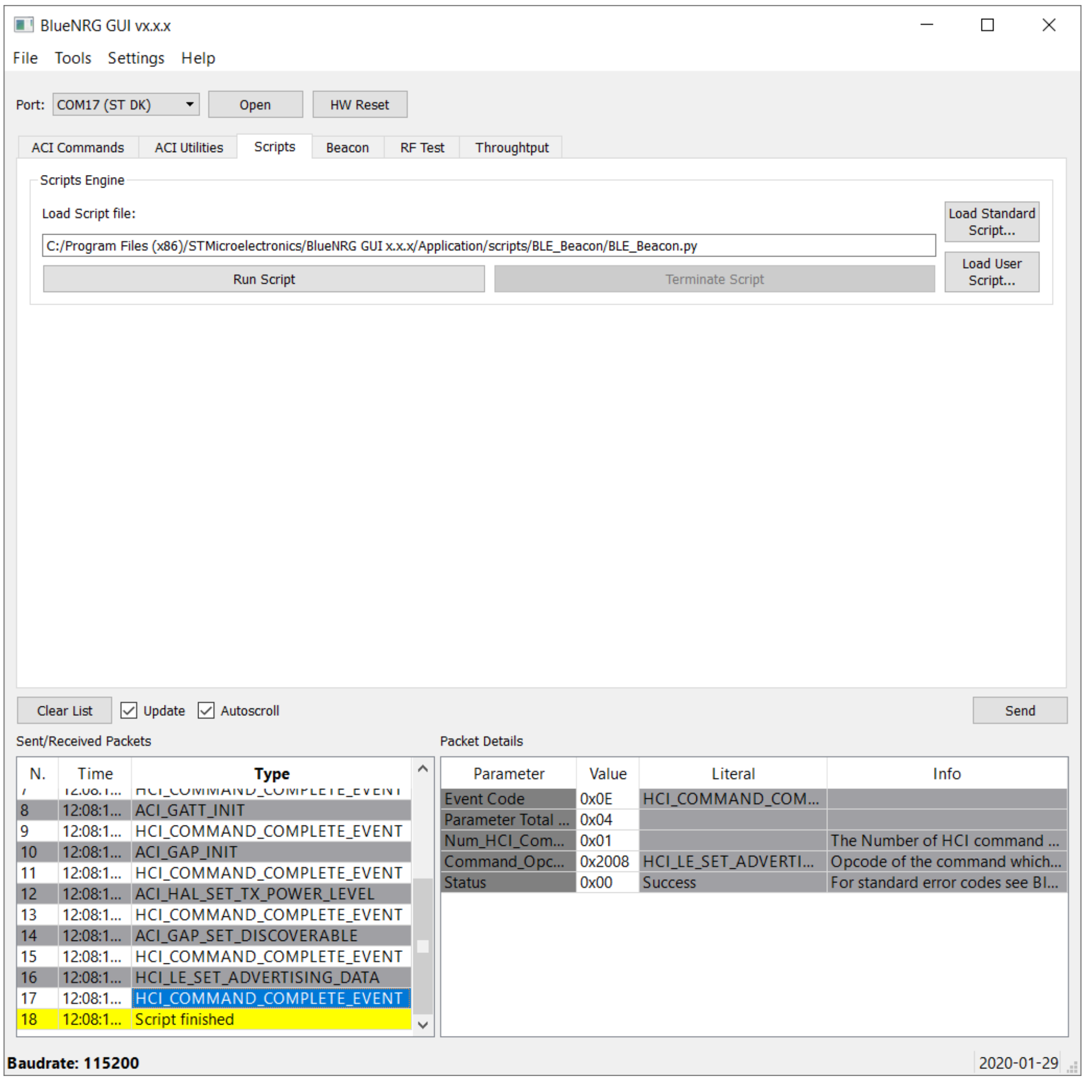
Table 3. **GUI ACI utility window: available peripheral operations**

Operation	Associated actions	Notes
Advertising	Allows placing a peripheral device in advertising mode by selecting: - Discoverable mode (limited, non-discoverable and general discoverable) - Type (ADV_IND, ADV_SCAN_IND, ADV_NONCONN_IND) - Set local name and type (complete or short) - Advertising intervals (min. and max.) - Policy: - Allows scan request from any, allows connect request from any - Allows scan request from white list only, allow connect request from any - Allows scan request from any, allows connect request from white list only	
Update advertising data	Allows updating the advertising data; Allows setting the scan response data; Allows updating the location UUID, major and minor number defined on the Beacon window	

2.2.4 GUI script window

The GUI script window allows the user to load and run a python script built using the available set of ACI commands and related events. For a list of supported HCI and ACI script commands and parameters, refer to those commands available in the GUI ACI command window.

Figure 8. **BlueNRG GUI script window section**



C:/Program Files (x86)/STMicroelectronics/BlueNRG GUI x.x.x/Application/scripts/BLE_Beacon/BLE_Beacon.py

Sent/Received Packets

N.	Time	Type
7	12:08:1...	HCI_COMMAND_COMPLETE_EVENT
8	12:08:1...	ACI_GATT_INIT
9	12:08:1...	HCI_COMMAND_COMPLETE_EVENT
10	12:08:1...	ACI_GAP_INIT
11	12:08:1...	HCI_COMMAND_COMPLETE_EVENT
12	12:08:1...	ACI_HAL_SET_TX_POWER_LEVEL
13	12:08:1...	HCI_COMMAND_COMPLETE_EVENT
14	12:08:1...	ACI_GAP_SET_DISCOVERABLE
15	12:08:1...	HCI_COMMAND_COMPLETE_EVENT
16	12:08:1...	HCI_LE_SET_ADVERTISING_DATA
17	12:08:1...	HCI_COMMAND_COMPLETE_EVENT
18	12:08:1...	Script finished

Packet Details

Parameter	Value	Literal	Info
Event Code	0x0E	HCI_COMMAND_COM...	
Parameter Total ...	0x04		
Num_HCI_Com...	0x01		The Number of HCI command ...
Command_Opc...	0x2008	HCI_LE_SET_ADVERTI...	Opcode of the command which...
Status	0x00	Success	For standard error codes see Bl...

Baudrate: 115200 2020-01-29

The script engine supports other utility commands listed in the following table.

Table 4. **GUI script window: utility commands**

Command name	Parameters	Description
ERROR	User message	Raises an exception with a user-defined debug message
CLEAR_QUEUE	None	Removes all the pending events collected within the internal event queue
GET_CHAR	None	Allows a specific char to be entered as input (as the C get_char() API)
GET_ALL_COM_PORT	None	Returns the list of all COM ports (ST DK kits and not)
GET_ALL_ST_DK_COM_PORT	None	Returns the list of all ST DK kits COM ports sorted by COM port number
GET_FILE	None	Allows a specific file to be selected as input
GET_NAME	None	Returns the device name within an advertising packet
GET_VALUE	Array of bytes	Converts the array of bytes to an integer value; e.g.,: X= [0x33,0x22] GET_VALUE(X) = 0x2233
GET_LIST	Integer, number of bytes	Converts the integer value to an array of number of bytes; e.g.,: X= 0x2233 GET_LIST(X, 2)= [0x33,0x22]
GET_STACK_VERSION	None	Returns the device information (HW version and FW version) as (hw, fw)
GET_RAND_KEY	None	Returns a random number between 0 and 999999
HW_BOOTLOADER	None	Hardware bootloader activation
HW_RESET	None	HW reset
INFO	String to be displayed	Opens a message window and show the input parameter. Script is blocked until user presses OK button
INSERT_FLOAT_NUMBER	None	Allows a float value
INSERT_INT_NUMBER	None	Allows an integer value to be entered
INSERT_PASS_KEY	None	Allows entering a pass key value for the security pass key method
IS_BlueNRG	None	Returns TRUE if the device is a BlueNRG, FALSE otherwise
IS_BlueNRG_MS	None	Returns TRUE if the device is a BlueNRG-MS, FALSE otherwise
IS_BlueNRG_1	None	Returns TRUE if the device is a BlueNRG-1, FALSE otherwise
IS_BlueNRG_2	None	Returns TRUE if the device is a BlueNRG-2, FALSE otherwise
PRINT	String	Print utility: displays information on GUI sent/received packets
RESET	None	SW reset
SLEEP	time	Sleeps for the period 'time' in ms
SET_MODE	Mode	Sets stack mode (1,2,3,4). Mode 4 is only supported on BlueNRG-MS from FW stack version 7.1b. Only for the BlueNRG, BlueNRG-MS devices.
SET_PUBLIC_ADDRESS	Public address	Set public address (optional)
SENSORDEMO_GET_TEMPERATURE	None	Allows retrieval of the temperature value from the ACI_ATT_READ_RESP_EVENT (only for the SensorDemo_Central script)
SENSORDEMO_GET_ACCELERATION	None	Allows retrieval of the acceleration values (x,y,z) from the ACI_GATT_NOTIFICATION_EVENT (only for the SensorDemo_Central script)
TIME	None	Returns the time as a floating point number expressed in seconds since the epoch, in UTC

The following pseudo code describes how to initialize a BlueNRG, BlueNRG-MS, BlueNRG-1 or BlueNRG-2 device as a peripheral using a simple python script:

```
#Reset device
HW_RESET()
#Init GATT
ACI_GATT_INIT()
#Init GAP as central device
ACI_GAP_INIT(Role=CENTRAL)
```

When a script calls a command which generates specific events, the script can detect them by using the WAIT_EVENT (event_code=None, timeout=None, continueOnEvtMiss=False, **param_checks) command.

Table 5. **WAIT_EVENT macro-command**

Command name	Description	Parameters	Return
WAIT_EVENT	It waits for an event with ‘Event Code’ parameter equal to event_code. If no event_code is indicated, the macro-command waits for any event. Optional filtering parameters allow additional filters to be defined on event fields	- event_code = none (default) - timeout = none (default) - continueOnEvtMiss = false (default) - param_checks = optional filtering parameters	- An event with its parameters None, if a timeout occurs and the input parameter “continueOnEvtMiss” is set to true - An event with its parameters - None, if a timeout occurs and the input parameter “continueOnEvtMiss” is set to true

The WAIT_EVENT macro-command waits for an event with 'Event Code' parameter equal to event_code. If no event_code is indicated, the macro-command waits for any event.

The timeout parameter allows the event timeout to be set. If no timeout is set, the macro-command awaits until an event occurs. If a timeout (greater than zero) is set and continueOnEvtMiss is false and no event occurs before the timeout, an HCITimeoutError error happens. Otherwise, if the input parameter continueOnEvtMiss is true and a timeout (greater than zero) is set, the macro-command returns the value none even when no event occurs before the timeout.

If one or more optional filtering parameters are specified, the macro-command performs a check on them and it returns only the first detected event that satisfies these parameters. The events received before the one returned are discarded.

The WAIT_EVENT() command return value can be:

- an event
- none, if a timeout occurs and the input parameter “continueOnEvtMiss” is set to true

An HCI timeout error exception is raised when a timeout occurs. Each coming event is stored in an internal queue. User is requested to call a specific WAIT_EVENT utility command in order to handle each received event and remove it from the internal queue. The CLEAR_QUEUE utility command allows the internal queue to be cleared in order to remove all the collected events not handled through the WAIT_EVENT utility command. The event_code parameter can be one of the following values:

Table 6. **Event codes with relative event parameter types**

event_code	Event parameter type	Event parameter type value
HCI_LE_META_EVENT	Subevent _Code	Refers to specific device BLE Stack ACI APIs and events doxygen documentation available on STSW-BNRGUI SW package, Docs Docs\gui_aci_html: ACI events, HCI LE meta events section
HCI_VENDOR_EVENT	Ecode	Refers to specific device BLE Stack ACI APIs and events doxygen documentation available on STSW-BNRGUI SW package, Docs Docs\gui_aci_html: ACI events, ACI GAP events ACI GATT/ATT events ACI L2CAP events ACI HAL event sections
NA	NA	Refers to specific device BLE Stack ACI APIs and events doxygen documentation available on STSW-BNRGUI SW package, Docs Docs\gui_aci_html: ACI events, HCI event section

Below are some code examples using the WAIT_EVENT() macro-command:

Example 1:

```
#Wait any events
evt = WAIT_EVENT()
if (evt.event_code == HCI_LE_META_EVENT):
     #User specific code ……
elif (evt.event_code==HCI_VENDOR_EVENT):
      #User specific code ……
```

Example 2:

```
# Wait an HCI_LE_META_EVENT
evt = WAIT_EVENT(HCI_LE_META_EVENT)
# Using evt.get_param('Subevent_Code').val it's possible to identify the specific HCI_LE_META
_EVENT
# parameter type value
evtCode = evt.get_param('Subevent_Code').val
# Check if received event is HCI_LE_CONNECTION_COMPLETE_EVENT
if (evtCode == HCI_LE_CONNECTION_COMPLETE_EVENT):
  # If Connection Complete Status is success, get connection handle
  if evt.get_param('Status').val==0x00:
    conn_handle= evt.get_param('Connection_Handle').val
```

Example 3:

```
#Wait HCI_VENDOR_EVENT event_code
evt = WAIT_EVENT(HCI_VENDOR_EVENT)
#Using evt.get_param('Ecode').val it's possible to identify the specific HCI_VENDOR_EVENT par
ameter type value
evtCode= evt.get_param('Ecode').val
if(evtCode == ACI_GATT_NOTIFICATION_EVENT):
    conn_handle=evt.get_param('Connection_Handle').val
```

Example 4:

```
#Wait the Ecode ACI_GATT_PROC_COMPLETE_EVENT (HCI_VENDOR_EVENT #event_code).
#if no event occurs within the selected timeout, an exception is raised
WAIT_EVENT(HCI_VENDOR_EVENT, timeout=30, Ecode=ACI_GATT_PROC_COMPLETE_EVENT)
```

Note: If no timeout parameter is specified, it awaits until the ACI_GATT_PROC_COMPLETE_EVENT.

Example 5:

```
#Wait an event for 10 seconds with continueOnEvtMiss set to True#
If no event occurs, the script continues (no exception is raised).
WAIT_EVENT(timeout=10, continueOnEvtMiss =True)
```

Note: If continueOnEvtMiss parameter is set to false and if no event within the selected timeout occurs, an exception is raised.

Example 6:

```
#Wait the HCI_DISCONNECTION_COMPLETE_EVENT event_code
WAIT_EVENT(HCI_DISCONNECTION_COMPLETE _EVENT)
```

Example 7:

```
#Create a Connection and wait for the HCI_LE_CONNECTION_COMPLETE_EVENT
ACI_GAP_CREATE_CONNECTION(Peer_Address=[0x12, 0x34, 0x00, 0xE1, 0x80, 0x02])
event= WAIT_EVENT(HCI_LE_META_EVENT, timeout=30,Subevent_Code=HCI_LE_CONNECTION_COMPLETE_EVEN
T)
if event.get_param('Status').val==0x00:
   # Store the connection handle
   conn_handle= event.get_param('Connection_Handle').val
   #User defined code…
```

GUI script engine loading and running steps

To load and run a python script using the BlueNRG GUI script engine, the following steps must be observed:

1. In the BlueNRG GUI, Scripts window, Script Engine section, click on tab "…", browse to the script location and select the script
2. Click on the "Run Script" tab to run the script. The execution flow (commands and events) is displayed in the BlueNRG GUI "Sent/Received Packets" section

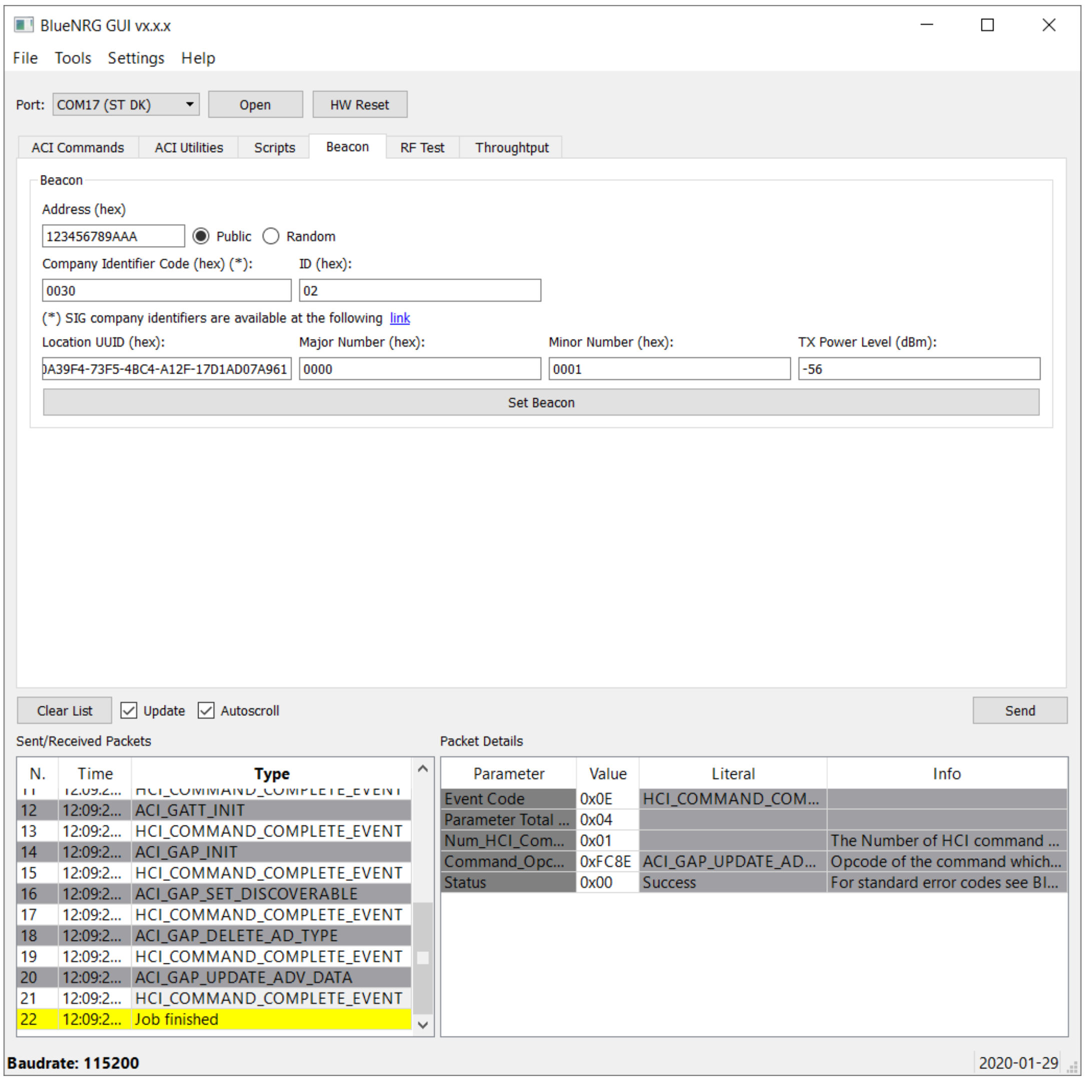
In the BlueNRG GUI SW package and future versions, some reference scripts are available in the GUI/scripts folder.

Note: To write and use the scripts, the user should have some knowledge of the Python language (Python 2.7.6), and a good understanding of the BLE stack ACI commands and related events.

2.2.5 GUI Beacon window

The BlueNRG GUI Beacon window provides some tabs to configure a BlueNRG, BlueNRG-MS, BlueNRG-1 or BlueNRG-2 device as a BLE Beacon device that transmits advertising packets with specific manufacturer data.

Figure 9. BlueNRG GUI Beacon window



Port: COM17 (ST DK)

Beacon

Address (hex): 123456789AAA [✓ Public] Random

Company Identifier Code (hex) (*): 0030

ID (hex): 02

(*) SIG company identifiers are available at the following link

Location UUID (hex):)A39F4-73F5-4BC4-A12F-17D1AD07A961

Major Number (hex): 0000

Minor Number (hex): 0001

TX Power Level (dBm): -56

☑ Update ☑ Autoscroll

Sent/Received Packets

N.	Time	Type
11	12:09:2...	HCI_COMMAND_COMPLETE_EVENT
12	12:09:2...	ACI_GATT_INIT
13	12:09:2...	HCI_COMMAND_COMPLETE_EVENT
14	12:09:2...	ACI_GAP_INIT
15	12:09:2...	HCI_COMMAND_COMPLETE_EVENT
16	12:09:2...	ACI_GAP_SET_DISCOVERABLE
17	12:09:2...	HCI_COMMAND_COMPLETE_EVENT
18	12:09:2...	ACI_GAP_DELETE_AD_TYPE
19	12:09:2...	HCI_COMMAND_COMPLETE_EVENT
20	12:09:2...	ACI_GAP_UPDATE_ADV_DATA
21	12:09:2...	HCI_COMMAND_COMPLETE_EVENT
22	12:09:2...	Job finished

Packet Details

Parameter	Value	Literal	Info
Event Code	0x0E	HCI_COMMAND_COM...	
Parameter Total ...	0x04		
Num_HCI_Com...	0x01		The Number of HCI command ...
Command_Opc...	0xFC8E	ACI_GAP_UPDATE_AD...	Opcode of the command which...
Status	0x00	Success	For standard error codes see Bl...

Baudrate: 115200

2020-01-29

The user can configure the advertising data fields in the following table for the BLE Beacon device through the BlueNRG GUI Beacon window configuration parameters.

Table 7. BlueNRG GUI Beacon window configuration parameters

Data field	Description	Notes
Address	Device address	
Public or random	Device address type	
Company identifier code	SIG company identifier	Default is 0x0030 (STMicroelectronics)
ID	Beacon ID	Fixed value
Location UUID	Beacons UUID	Used to distinguish specific Beacons from others
Major number	Identifier for a group of beacons	Used to group a related set of Beacons
Minor number	Identifier for a single beacon	Used to identify a single Beacon
TxPower Level	2's complement of the Tx power	Used to establish how far you are from the device

To configure the selected platform as a BLE beacon device, click on the “Set Beacon” tab.

2.2.6 GUI RF test window

The BlueNRG GUI provides the RF test window that permits to perform the following tests:

1. Start/stop a tone on a specific BLE RF channel
2. Perform a BLE packer error rate (PER) tests using BLE direct test mode (DTM) commands

Start/stop a tone

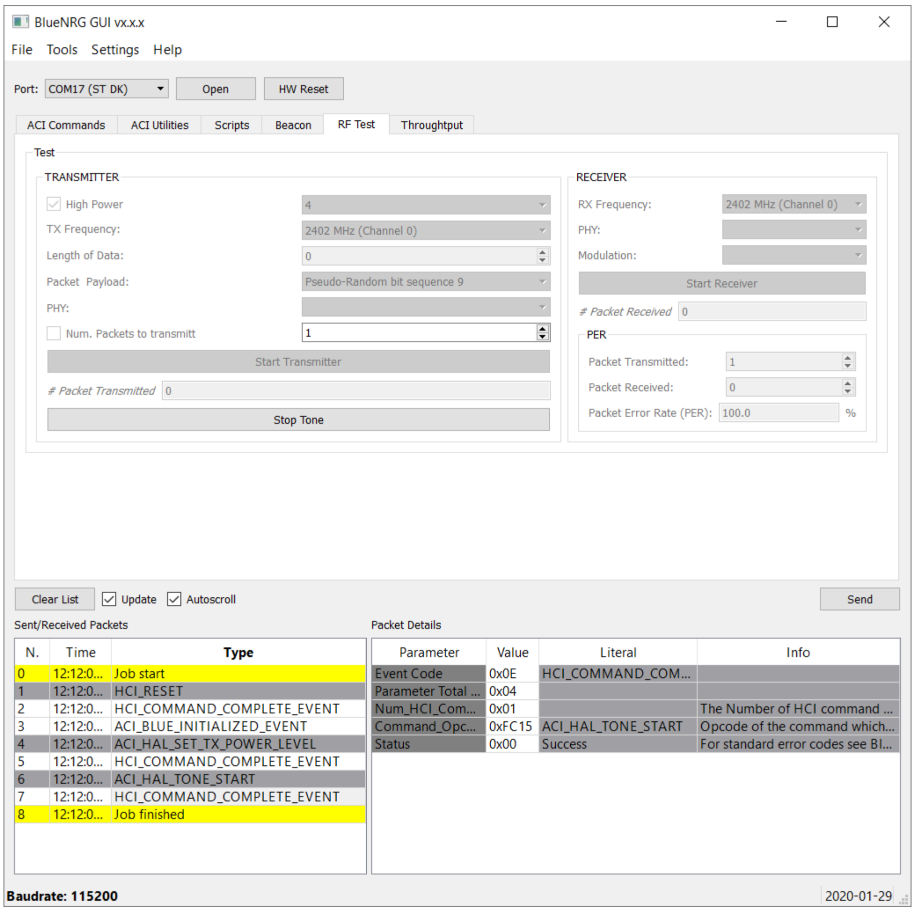
To start a tone on a specific RF BLE channel:

1. Connect a BlueNRG, BlueNRG-MS, BlueNRG-1 or BlueNRG-2 platform to PC USB port
2. Launch an instance of BlueNRG GUI
3. Open the relative COM port
4. Go to RF test window and in the TRANSMITTER section:
   a. Set BLE channel by TX frequency combo box
   b. Set TX power in the related combo box
   c. Click on the "Start Tone" button

To stop a tone on a specific RF BLE channel:

1. Go to RF test window and in the TRANSMITTER section:
   a. Click on stop tone button (stop button is available only when a tone is started)

Figure 10. GUI RF test: start a tone



Port: COM17 (ST DK)

TRANSMITTER

High Power: 4
TX Frequency: 2402 MHz (Channel 0)
Length of Data: 0
Packet Payload: Pseudo-Random bit sequence 9
PHY:
Num. Packets to transmitt: 1
# Packet Transmitted 0

RECEIVER

RX Frequency: 2402 MHz (Channel 0)
PHY:
Modulation:
# Packet Received 0

PER

Packet Transmitted: 1
Packet Received: 0
Packet Error Rate (PER): 100.0 %

Sent/Received Packets

N.	Time	Type
0	12:12:0...	Job start
1	12:12:0...	HCI_RESET
2	12:12:0...	HCI_COMMAND_COMPLETE_EVENT
3	12:12:0...	ACI_BLUE_INITIALIZED_EVENT
4	12:12:0...	ACI_HAL_SET_TX_POWER_LEVEL
5	12:12:0...	HCI_COMMAND_COMPLETE_EVENT
6	12:12:0...	ACI_HAL_TONE_START
7	12:12:0...	HCI_COMMAND_COMPLETE_EVENT
8	12:12:0...	Job finished

Packet Details

Parameter	Value	Literal	Info
Event Code	0x0E	HCI_COMMAND_COM...	
Parameter Total ...	0x04		
Num_HCI_Com...	0x01		The Number of HCI command ...
Command_Opc...	0xFC15	ACI_HAL_TONE_START	Opcode of the command which...
Status	0x00	Success	For standard error codes see Bl...

Baudrate: 115200 2020-01-29

Direct test mode (DTM) tests

The BlueNRG GUI RF test let you target a packet error rate test scenario using the BLE direct test mode commands.

Two sections are available:

1. TRANSMITTER section to transmit reference packets to a fixed interval
2. RECEIVER section to receive reference packets to a fixed interval

TRANSMITTER section

This section let you set the following items:

- The power level of transmitter
- The frequency of transmitter
- Length of data to transmit in each packet
- Packet payload format as defined on Bluetooth low energy specification, direct test mode section

Clicking on "Start Transmitter" button, test reference packets are sent to a fixed interval.

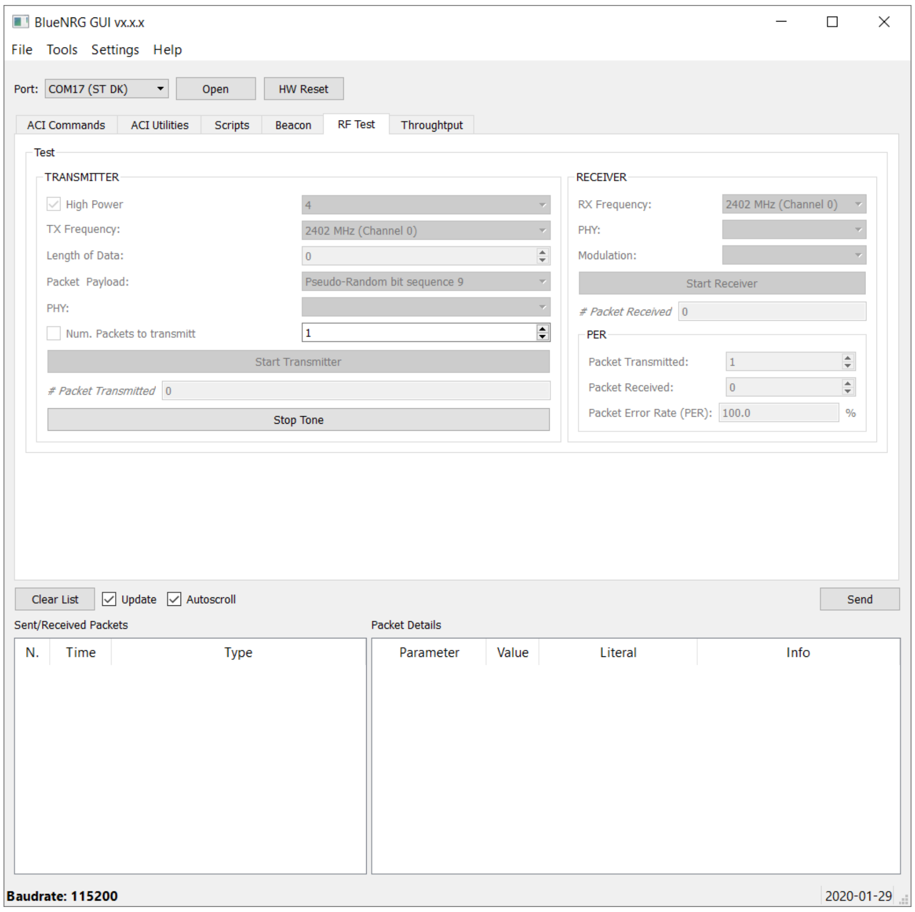
RECEIVER section

This section let you set the following items:

- The frequency of receiver

Clicking on "Start Receiver" button, test reference packets are received to a fixed interval.

Figure 11. GUI RF test: transmitter and receiver sections



Port: COM17 (ST DK)

High Power: 4
TX Frequency: 2402 MHz (Channel 0)
Length of Data: 0
Packet Payload: Pseudo-Random bit sequence 9
PHY:
Num. Packets to transmitt: 1
# Packet Transmitted 0

RX Frequency: 2402 MHz (Channel 0)
PHY:
Modulation:
# Packet Received 0

Packet Transmitted: 1
Packet Received: 0
Packet Error Rate (PER): 100.0 %

Baudrate: 115200
2020-01-29

Packet error rate (PER) test procedure

To perform a packet error rate test using the standard BLE direct test mode commands (HCI_LE_Transmitter_Test, HCI_LE_Receiver_Test and HCI_LE_Test_End), the following steps are needed:

Start PER test

1. Connect two platforms (BlueNRG, BlueNRG-MS, BlueNRG-1 or BlueNRG-2) to PC USB ports
2. Open two instances of BlueNRG GUI on both devices (called TX and RX devices, respectively)
3. In each instance of BlueNRG GUI, open COM port relative to the TX/RX device
4. Ensure the antennas are plugged into the devices where applicable
5. In the GUI related to RX device:
   – Go to RF test window, RECEIVER section
   – Set RX frequency
   – Click on "Start Receiver" button to start receiver test
6. In the GUI related to TX device:
   – Go to RF test window, TRANSMITTER section
   – Set TX power
   – Set TX frequency
   – Set length of data
   – Set packet payload format
   – Click on "Start Transmitter" button, to start transmitter test

Stop PER test

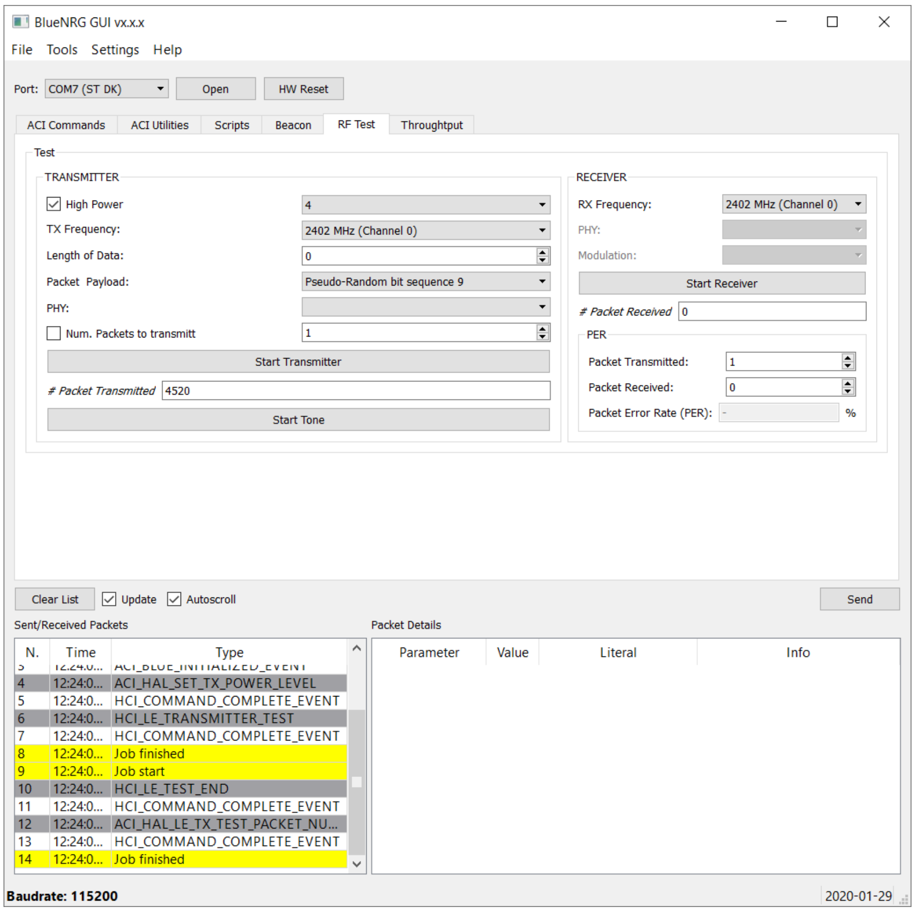
1. In the GUI related to TX device:
   – Go to RF test window, TRANSMITTER section
   – Click on "Stop Transmitter" button. The number of transmitted packets are displayed on #Packet Transmitted field.
2. In the GUI related to RX device
   – Go to RF test window, RECEIVER section
   – Click on "Stop Receiver" button. The number of received packets are displayed on #PacketReceived field.

Get PER (packet error rate) value

1. In the GUI related to RX device:

- Go to RF test window, RECEIVER section
- In the PER section, insert the number of transmitted packet from TX device in the packet transmitted field (read this value from TRANSMITTER section in the GUI related to TX device)
- The PER (packet error rate) value is showed in the packet error rate field

Figure 12. GUI RF test, PER test: TX device



BlueNRG GUI vx.x.x

Port: COM7 (ST DK)

Test

TRANSMITTER

- High Power: ✓ 4
- TX Frequency: 2402 MHz (Channel 0)
- Length of Data: 0
- Packet Payload: Pseudo-Random bit sequence 9
- PHY:
- Num. Packets to transmitt: ☐ 1

# Packet Transmitted 4520

RECEIVER

- RX Frequency: 2402 MHz (Channel 0)
- PHY:
- Modulation:

# Packet Received 0

PER

- Packet Transmitted: 1
- Packet Received: 0
- Packet Error Rate (PER): - %

Update ✓ Autoscroll ✓

Sent/Received Packets

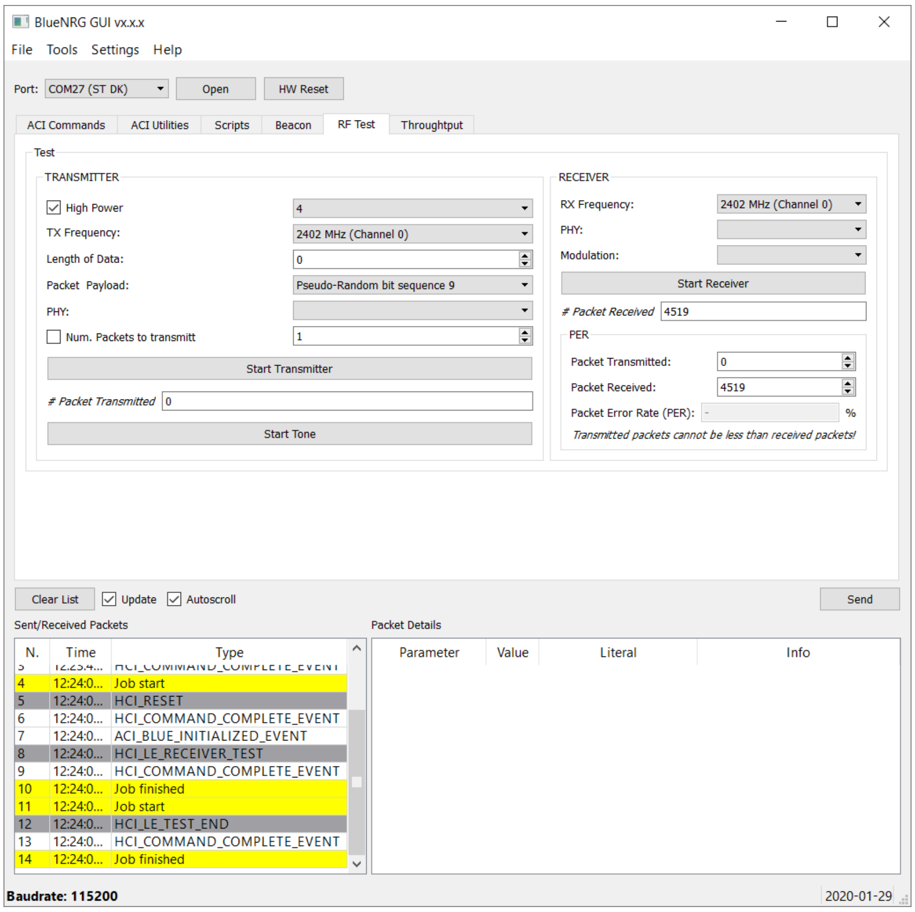
N.	Time	Type
3	12:24:0...	ACI_BLUE_INITIALIZED_EVENT
4	12:24:0...	ACI_HAL_SET_TX_POWER_LEVEL
5	12:24:0...	HCI_COMMAND_COMPLETE_EVENT
6	12:24:0...	HCI_LE_TRANSMITTER_TEST
7	12:24:0...	HCI_COMMAND_COMPLETE_EVENT
8	12:24:0...	Job finished
9	12:24:0...	Job start
10	12:24:0...	HCI_LE_TEST_END
11	12:24:0...	HCI_COMMAND_COMPLETE_EVENT
12	12:24:0...	ACI_HAL_LE_TX_TEST_PACKET_NU...
13	12:24:0...	HCI_COMMAND_COMPLETE_EVENT
14	12:24:0...	Job finished

Packet Details

Parameter	Value	Literal	Info

Baudrate: 115200 2020-01-29

Figure 13. **GUI RF Test, PER test: RX device**



Port: COM27 (ST DK)

Test

TRANSMITTER

☑ High Power	4
TX Frequency:	2402 MHz (Channel 0)
Length of Data:	0
Packet Payload:	Pseudo-Random bit sequence 9
PHY:	
☐ Num. Packets to transmitt	1

# Packet Transmitted 0

RECEIVER

RX Frequency:	2402 MHz (Channel 0)
PHY:	
Modulation:	

# Packet Received 4519

PER

Packet Transmitted:	0
Packet Received:	4519
Packet Error Rate (PER):	- %

Transmitted packets cannot be less than received packets!

☑ Update ☑ Autoscroll

Sent/Received Packets

N.	Time	Type
3	12:23:4...	HCI_COMMAND_COMPLETE_EVENT
4	12:24:0...	Job start
5	12:24:0...	HCI_RESET
6	12:24:0...	HCI_COMMAND_COMPLETE_EVENT
7	12:24:0...	ACI_BLUE_INITIALIZED_EVENT
8	12:24:0...	HCI_LE_RECEIVER_TEST
9	12:24:0...	HCI_COMMAND_COMPLETE_EVENT
10	12:24:0...	Job finished
11	12:24:0...	Job start
12	12:24:0...	HCI_LE_TEST_END
13	12:24:0...	HCI_COMMAND_COMPLETE_EVENT
14	12:24:0...	Job finished

Packet Details

Parameter	Value	Literal	Info

Baudrate: 115200 2020-01-29

Note: Starting from the STSW-BNRGUI v3.2.0 or later, BlueNRG-1,2 BLE stack v2.x, if user specifies the number of packets to be transmitted on a related field (number of packets to transmit), the ACI_HAL_TRANSMITTER_TEST_PACKETS API is used to perform the transmitter test. This API is a customized version of the HCI_LE_TRANSMITTER_TEST API which allows the number of packets to be specified and to be sent. Please notice that user must uncheck the field Num. Packets to transmit in order to use the standard HCI_LE_TRANSMITTER_TEST API.

2.2.7 GUI RF throughput window

The BlueNRG GUI RF throughput window allows throughput tests to be performed by setting up a BLE communication between two devices, master and slaves, and setting up one of the following scenarios:

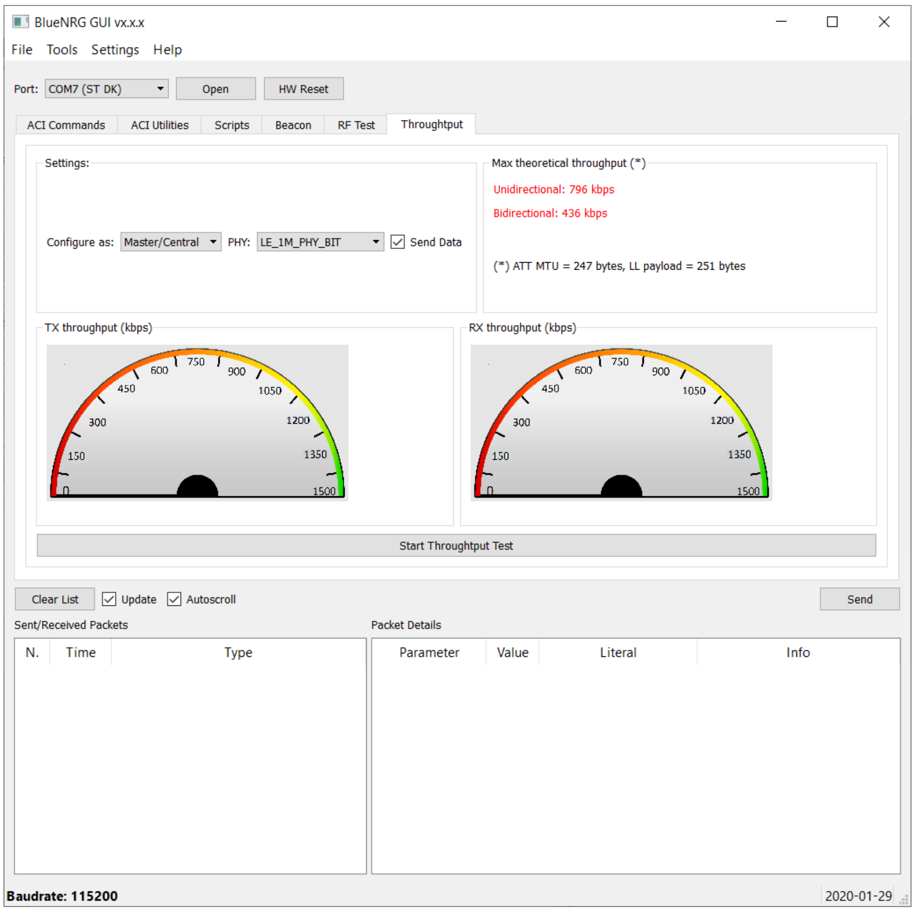
1. Unidirectional communication (characteristic notification) from slave (GATT server) to master (GATT client) device
2. Unidirectional communication (write without response) from master (GATT client) to slave (GATT server) device
3. Bidirectional communication (characteristic notification) from slave (GATT server) to master (GATT client) device and (write without response) from master (GATT client) to slave (GATT server) device

The theoretical throughput is calculated taking in account the ATT MTU = 247 bytes and LL payload = 251 bytes for the BlueNRG-2 device (it supports data length extension feature) and ATT MTU = 247 bytes and LL payload = 27 bytes for the BlueNRG-1 device.

This feature supports the BlueNRG-1, BlueNRG-2 devices only with BLE firmware stack v2.x.

Note: 1. *This feature requires the BlueNRG-1.2 devices to be loaded with a dedicated DTM binary image (DTM_UART_Throughput.hex) stored on firmware/{BlueNRG-1|BlueNRG-2}/DTM folders (serial baudrate is 115200)*

Figure 14. BlueNRG GUI RF throughput window



Port: COM7 (ST DK)

Configure as: Master/Central PHY: LE_1M_PHY_BIT ✓ Send Data

Unidirectional: 796 kbps
Bidirectional: 436 kbps

(*) ATT MTU = 247 bytes, LL payload = 251 bytes

Baudrate: 115200   2020-01-29

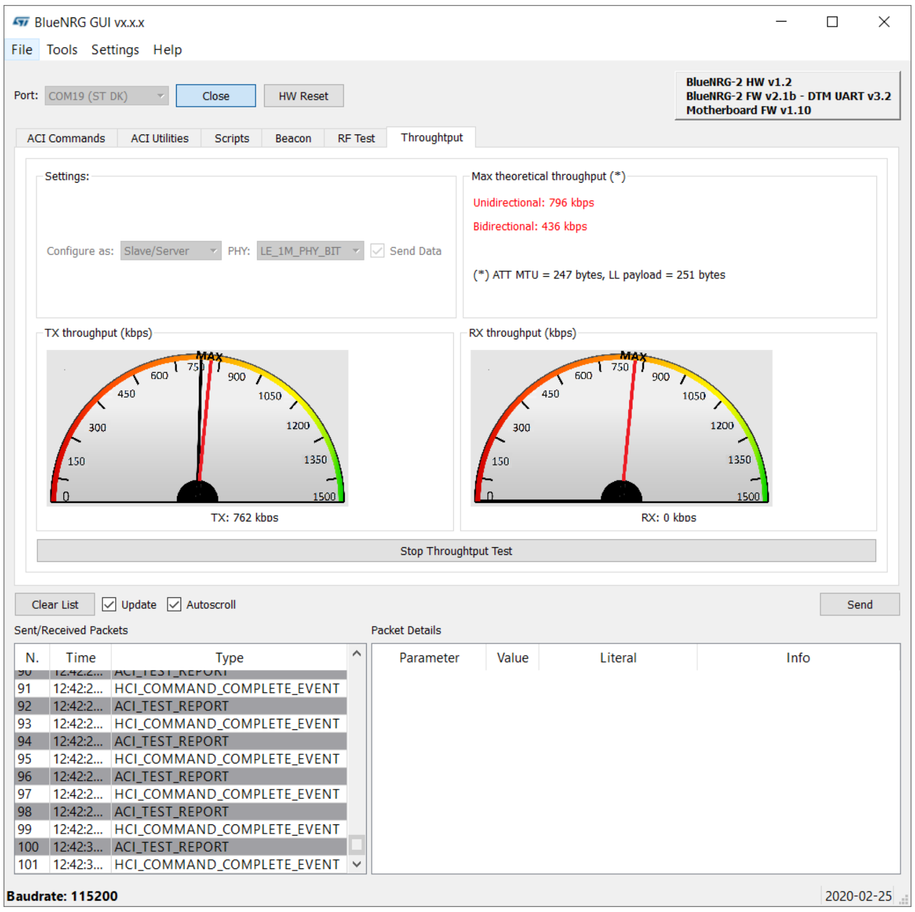
Unidirectional (slave to master) throughput test

1. Connect two platforms (BlueNRG-2) to PC USB ports
2. Open two instances of the BlueNRG GUI on both devices (called master and slave devices, respectively)
3. In each instance of the BlueNRG GUI, open COM port relative to the master/slave device
4. Ensure the antennas are plugged into the devices, where applicable

Slave device

1. On Settings, Configure As: select Slave/Server and set Send data checkbox
2. On Settings, select Send Data to enable communication from slave to master device (characteristic notification)
3. Click on "Start Throughput" button to start throughput test:
    a. Slave device enters discoverable mode
    b. Once Master device connects to Slave device, the actual throughput value (numeric value and black line) and the theoretical expected throughput value (numeric value and red line) are displayed on Slave TX window and on Master RX window
4. Click on “Stop throughput” to stop the test

Figure 15. **BlueNRG GUI RF throughput unidirectional, slave sends data**



Port: COM19 (ST DK)

BlueNRG-2 HW v1.2
BlueNRG-2 FW v2.1b - DTM UART v3.2
Motherboard FW v1.10

Configure as: Slave/Server PHY: LE_1M_PHY_BIT ✓ Send Data

Max theoretical throughput (*)
Unidirectional: 796 kbps
Bidirectional: 436 kbps
(*) ATT MTU = 247 bytes, LL payload = 251 bytes

TX: 762 kbps

RX: 0 kbps

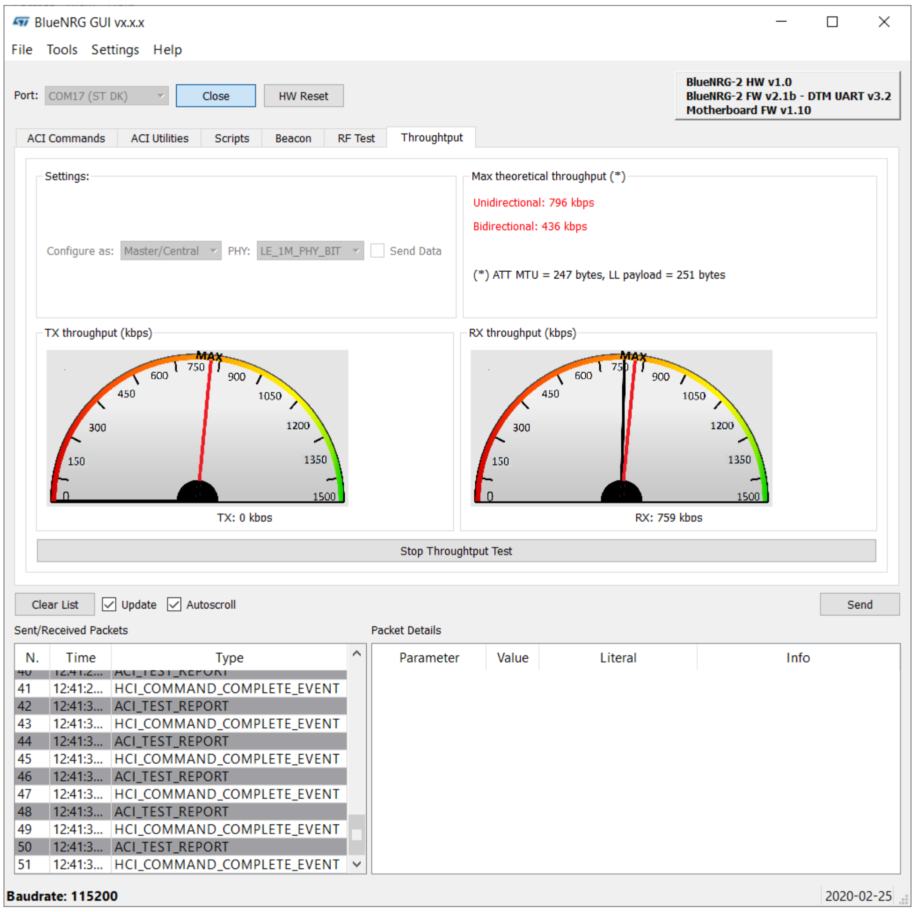
N.	Time	Type
90	12:42:2...	ACI_TEST_REPORT
91	12:42:2...	HCI_COMMAND_COMPLETE_EVENT
92	12:42:2...	ACI_TEST_REPORT
93	12:42:2...	HCI_COMMAND_COMPLETE_EVENT
94	12:42:2...	ACI_TEST_REPORT
95	12:42:2...	HCI_COMMAND_COMPLETE_EVENT
96	12:42:2...	ACI_TEST_REPORT
97	12:42:2...	HCI_COMMAND_COMPLETE_EVENT
98	12:42:2...	ACI_TEST_REPORT
99	12:42:2...	HCI_COMMAND_COMPLETE_EVENT
100	12:42:3...	ACI_TEST_REPORT
101	12:42:3...	HCI_COMMAND_COMPLETE_EVENT

Baudrate: 115200 2020-02-25

Master device

1. On settings, configure as: select Master/Central and unset Send Data checkbox
2. Click on "Start Throughput" button to start throughput test:
    a. Master connects to slave device and then it enables characteristic notifications: slave device starts notifications flow
    b. BLE communication from slave to master (characteristic notification) is started.The actual throughput value (numeric value and black line) and the theoretical expected throughput value (numeric value and red line) are displayed on Slave TX window and on Master RX window.
3. Click on “Stop throughput” to stop the test

Figure 16. **BlueNRG GUI RF throughput unidirectional, master receives data**



BlueNRG GUI vx.x.x

File Tools Settings Help

Port: COM17 (ST DK) Close HW Reset

BlueNRG-2 HW v1.0
BlueNRG-2 FW v2.1b - DTM UART v3.2
Motherboard FW v1.10

ACI Commands | ACI Utilities | Scripts | Beacon | RF Test | Throughput

Settings:

Configure as: Master/Central PHY: LE_1M_PHY_BIT Send Data

Max theoretical throughput (*)

Unidirectional: 796 kbps

Bidirectional: 436 kbps

(*) ATT MTU = 247 bytes, LL payload = 251 bytes

TX throughput (kbps)

TX: 0 kbps

RX throughput (kbps)

RX: 759 kbps

Stop Throughput Test

Clear List ✓ Update ✓ Autoscroll Send

Sent/Received Packets

N.	Time	Type
41	12:41:2...	HCI_COMMAND_COMPLETE_EVENT
42	12:41:3...	ACI_TEST_REPORT
43	12:41:3...	HCI_COMMAND_COMPLETE_EVENT
44	12:41:3...	ACI_TEST_REPORT
45	12:41:3...	HCI_COMMAND_COMPLETE_EVENT
46	12:41:3...	ACI_TEST_REPORT
47	12:41:3...	HCI_COMMAND_COMPLETE_EVENT
48	12:41:3...	ACI_TEST_REPORT
49	12:41:3...	HCI_COMMAND_COMPLETE_EVENT
50	12:41:3...	ACI_TEST_REPORT
51	12:41:3...	HCI_COMMAND_COMPLETE_EVENT

Packet Details

Parameter	Value	Literal	Info

Baudrate: 115200 2020-02-25

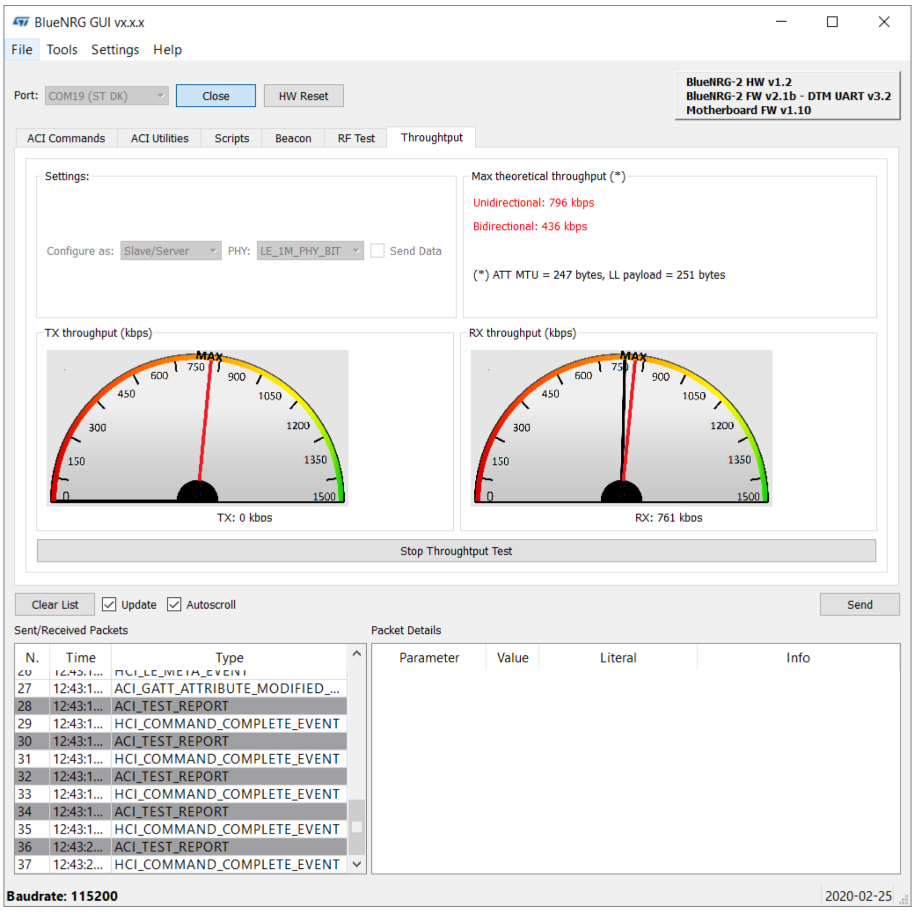
Unidirectional (master-to-slave) throughput test

1. Connect two platforms (BlueNRG-2) to PC USB ports
2. Open two instances of the BlueNRG GUI on both devices (called master and slave devices, respectively)
3. In each instance of the BlueNRG GUI, open COM port relative to the master/slave device
4. Ensure the antennas are plugged into the devices where applicable

Slave device

1. On Settings, Configure As: select Slave/Server and unset Send Data checkbox
2. Click on "Start Throughput" button to start throughput test:
   a. Slave device enters discoverable mode
   b. Once master device connects to slave device, the actual throughput value (numeric value and black line) and the theoretical expected throughput value (numeric value and red line) are displayed on slave RX window and on master TX window
3. Click on “Stop throughput” to stop the test

Figure 17. **BlueNRG GUI RF throughput unidirectional:slave receives data**



BlueNRG GUI vx.x.x

File Tools Settings Help

Port: COM19 (ST DK) Close HW Reset

BlueNRG-2 HW v1.2
BlueNRG-2 FW v2.1b - DTM UART v3.2
Motherboard FW v1.10

ACI Commands | ACI Utilities | Scripts | Beacon | RF Test | Throughtput

Settings:

Configure as: Slave/Server PHY: LE_1M_PHY_BIT Send Data

Max theoretical throughput (*)

Unidirectional: 796 kbps

Bidirectional: 436 kbps

(*) ATT MTU = 247 bytes, LL payload = 251 bytes

TX throughput (kbps) — TX: 0 kbps

RX throughput (kbps) — RX: 761 kbps

Stop Throughput Test

Clear List | Update | Autoscroll | Send

Sent/Received Packets

N.	Time	Type
26	12:43:1...	HCI_LE_META_EVENT
27	12:43:1...	ACI_GATT_ATTRIBUTE_MODIFIED_...
28	12:43:1...	ACI_TEST_REPORT
29	12:43:1...	HCI_COMMAND_COMPLETE_EVENT
30	12:43:1...	ACI_TEST_REPORT
31	12:43:1...	HCI_COMMAND_COMPLETE_EVENT
32	12:43:1...	ACI_TEST_REPORT
33	12:43:1...	HCI_COMMAND_COMPLETE_EVENT
34	12:43:1...	ACI_TEST_REPORT
35	12:43:1...	HCI_COMMAND_COMPLETE_EVENT
36	12:43:2...	ACI_TEST_REPORT
37	12:43:2...	HCI_COMMAND_COMPLETE_EVENT

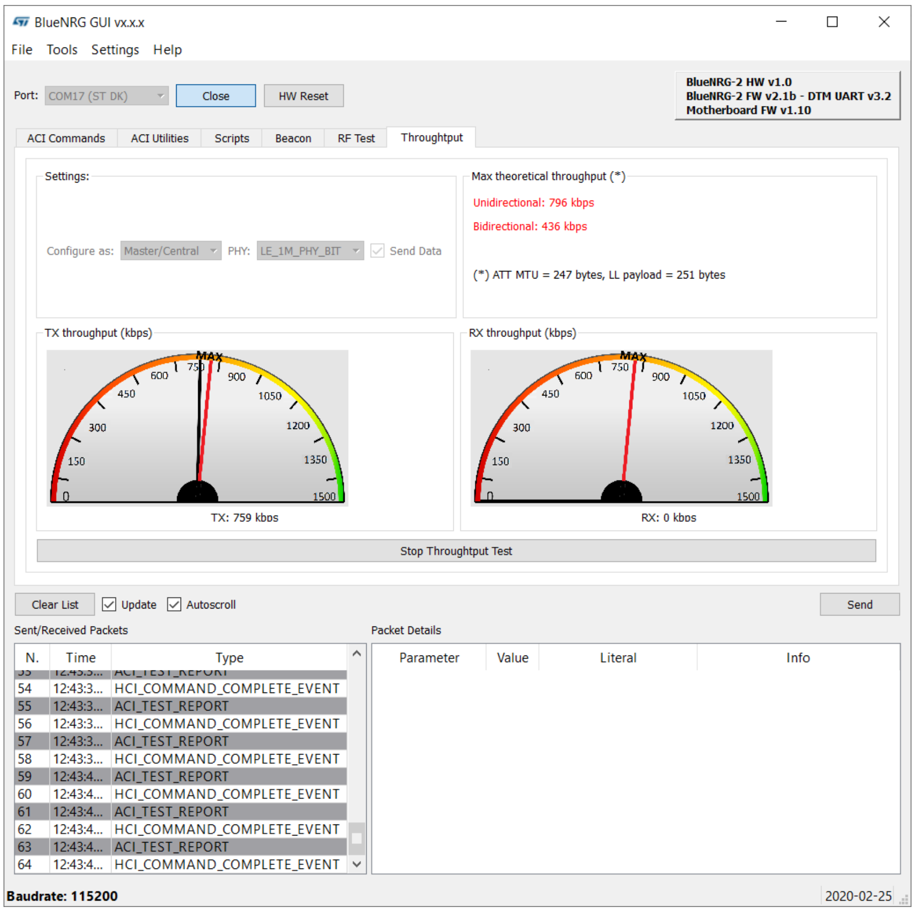
Packet Details

Parameter	Value	Literal	Info

Baudrate: 115200 2020-02-25

Master device

1. On Settings, Configure As: select Master/Central and set Send Data checkbox
2. On Settings, select Send Data to enable communication from master to slave device (characteristic write without response)
3. Click on "Start Throughput" button to start throughput test:
   a. Master connects to slave device
   b. BLE communication from master to slave (characteristic write without response) is started. Once master device connects to slave device, the actual throughput value (numeric value and black line) and the theoretical expected throughput value (numeric value and red line) are displayed on slave RX window and on master TX window
4. Click on “Stop throughput” to stop the test.



Port: COM17 (ST DK)

BlueNRG-2 HW v1.0
BlueNRG-2 FW v2.1b - DTM UART v3.2
Motherboard FW v1.10

Settings:

Configure as: Master/Central PHY: LE_1M_PHY_BIT ☐ Send Data

Max theoretical throughput (*)

Unidirectional: 796 kbps

Bidirectional: 436 kbps

(*) ATT MTU = 247 bytes, LL payload = 251 bytes

TX throughput (kbps)

TX: 759 kbps

RX throughput (kbps)

RX: 0 kbps

☑ Update ☑ Autoscroll

Sent/Received Packets

N.	Time	Type
53	12:43:3...	ACI_TEST_REPORT
54	12:43:3...	HCI_COMMAND_COMPLETE_EVENT
55	12:43:3...	ACI_TEST_REPORT
56	12:43:3...	HCI_COMMAND_COMPLETE_EVENT
57	12:43:3...	ACI_TEST_REPORT
58	12:43:3...	HCI_COMMAND_COMPLETE_EVENT
59	12:43:4...	ACI_TEST_REPORT
60	12:43:4...	HCI_COMMAND_COMPLETE_EVENT
61	12:43:4...	ACI_TEST_REPORT
62	12:43:4...	HCI_COMMAND_COMPLETE_EVENT
63	12:43:4...	ACI_TEST_REPORT
64	12:43:4...	HCI_COMMAND_COMPLETE_EVENT

Packet Details

Parameter	Value	Literal	Info

Baudrate: 115200

2020-02-25

Figure 18. **BlueNRG GUI RF throughput bidirectional:master sends data**

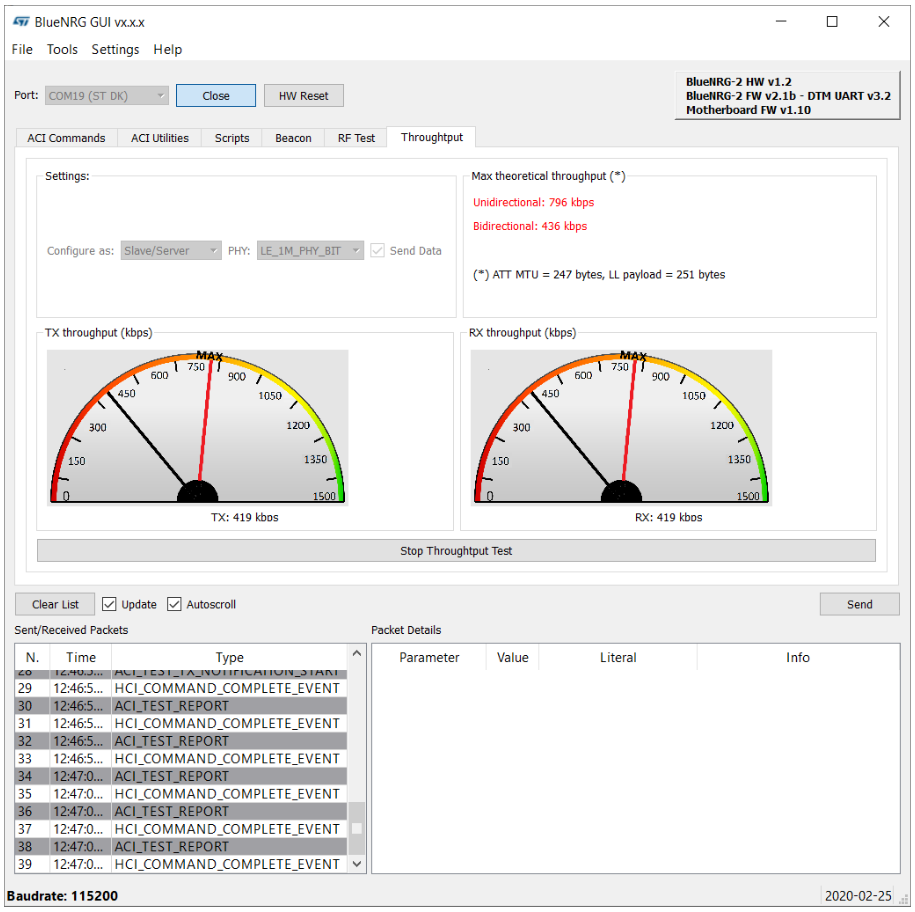
Bidirectional (slave-to-master, master-to-slave) throughput test

1. Connect two platforms (BlueNRG-2) to PC USB ports
2. Open two instances of the BlueNRG GUI on both devices (called master and slave devices, respectively)
3. In each instance of the BlueNRG GUI, open COM port relative to the master/slave device
4. Ensure the antennas are plugged into the devices where applicable

Slave device

1. On Settings, Configure As: select Slave/Server
2. On Settings, select Send Data to enable communication from slave-to-master device (characteristic notification)
3. Click on "Start Throughput" button to start throughput test:
   a. Slave device enters discoverable mode
   b. Once master device connects to slave device, the actual throughput value (numeric value and black line) and the theoretical expected throughput value (numeric value and red line) are displayed on slave master TX, RX windows
4. Click on "Start Throughput" button to start throughput test

Figure 19. **BlueNRG GUI RF throughput bidirectional:slave sends data**



Port: COM19 (ST DK)

BlueNRG-2 HW v1.2
BlueNRG-2 FW v2.1b - DTM UART v3.2
Motherboard FW v1.10

Settings:

Configure as: Slave/Server PHY: LE_1M_PHY_BIT ✓ Send Data

Max theoretical throughput (*)

Unidirectional: 796 kbps

Bidirectional: 436 kbps

(*) ATT MTU = 247 bytes, LL payload = 251 bytes

TX throughput (kbps)

TX: 419 kbps

RX throughput (kbps)

RX: 419 kbps

✓ Update ✓ Autoscroll

Sent/Received Packets

N.	Time	Type
28	12:46:5...	ACI_TEST_TX_NOTIFICATION_START
29	12:46:5...	HCI_COMMAND_COMPLETE_EVENT
30	12:46:5...	ACI_TEST_REPORT
31	12:46:5...	HCI_COMMAND_COMPLETE_EVENT
32	12:46:5...	ACI_TEST_REPORT
33	12:46:5...	HCI_COMMAND_COMPLETE_EVENT
34	12:47:0...	ACI_TEST_REPORT
35	12:47:0...	HCI_COMMAND_COMPLETE_EVENT
36	12:47:0...	ACI_TEST_REPORT
37	12:47:0...	HCI_COMMAND_COMPLETE_EVENT
38	12:47:0...	ACI_TEST_REPORT
39	12:47:0...	HCI_COMMAND_COMPLETE_EVENT

Packet Details

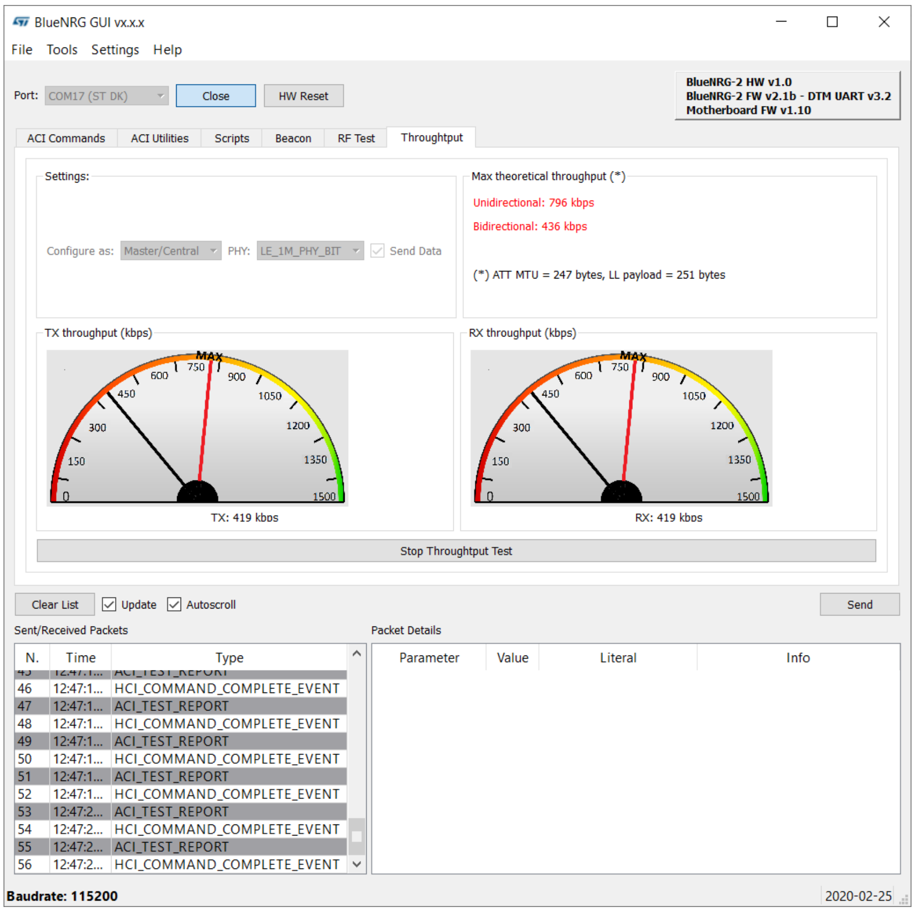
Baudrate: 115200

2020-02-25

Master device

1. On Settings, Configure As: select Master/Central and set Send Data checkbox
2. On Settings, select Send Data to enable communication from master to slave device (characteristic write without response)
3. Click on "Start Throughput" button to start throughput test:
   a. Master connects to slave device and then it enables characteristic notifications: slave device starts notifications flow
   b. BLE communication from slave to master (characteristic notification) is started. The actual throughput value is displayed the actual throughput value (numeric value and black line) and the theoretical expected throughput value (numeric value and red line) are displayed on slave, master TX, RX windows
4. Click on “Stop throughput” to stop the test

Figure 20. **BlueNRG GUI RF throughput bidirectional:master sends data**



BlueNRG-2 HW v1.0
BlueNRG-2 FW v2.1b - DTM UART v3.2
Motherboard FW v1.10

Configure as: Master/Central PHY: LE_1M_PHY_BIT ✓ Send Data

Max theoretical throughput (*)
Unidirectional: 796 kbps
Bidirectional: 436 kbps
(*) ATT MTU = 247 bytes, LL payload = 251 bytes

TX: 419 kbps

RX: 419 kbps

N.	Time	Type
46	12:47:1...	HCI_COMMAND_COMPLETE_EVENT
47	12:47:1...	ACI_TEST_REPORT
48	12:47:1...	HCI_COMMAND_COMPLETE_EVENT
49	12:47:1...	ACI_TEST_REPORT
50	12:47:1...	HCI_COMMAND_COMPLETE_EVENT
51	12:47:1...	ACI_TEST_REPORT
52	12:47:1...	HCI_COMMAND_COMPLETE_EVENT
53	12:47:2...	ACI_TEST_REPORT
54	12:47:2...	HCI_COMMAND_COMPLETE_EVENT
55	12:47:2...	ACI_TEST_REPORT
56	12:47:2...	HCI_COMMAND_COMPLETE_EVENT

Parameter	Value	Literal	Info

Baudrate: 115200 2020-02-25

3 Script launcher utility

The script launcher is a standalone utility, which let you run a script without using the BlueNRG GUI script engine tool.

The script launcher utility (BlueNRG_Script_Launcher.exe) is included in the BlueNRG GUI software package in the application folder.

3.1 Requirements

To use the script launcher utility on a specific device, the corresponding platform must be connected to a PC USB port and loaded with the same pre-built binary file used for the BlueNRG GUI, as described in Section 2.1 Requirements (BlueNRG_VCOM_x_x.hex for BlueNRG, BlueNRG-MS platforms and specific DTM binary files for the BlueNRG-1 and BlueNRG-2 platforms).

3.2 Script launcher utility options

To use the script launcher utility on a specific device, open a window DOS shell and launch the relative BlueNRG_Script_Launcher.exe (type –h to list all the supported options):

C:\Program Files (x86)\STMicroelectronics\BlueNRG GUI x.x.x\Application → BlueNRG_Script_Launcher.exe -h:

- -h, --help show this help message and exit
- -v, --version show program's version number and exit
- -g, --get get list of existing COM ports and exit
- -d, --debug debug script file (python pdb)
- -l, --log log data
- -p PORT, --port PORT select COM port
- -b BAUD_RATE, --baud rate BAUD_RATE set Baud Rate:
- -s SCRIPT_FILE, --script SCRIPT_FILE set script file name
- -z, --save save log in a file (*.csv, *.txt)

Option to get the list of all available COMx ports (devices are connected to the PC USB)

C:\Program Files (x86)\STMicroelectronics\BlueNRG GUI GUI x.x.x\Application BlueNRG_Script_Launcher.exe -g

List of available COM ports:

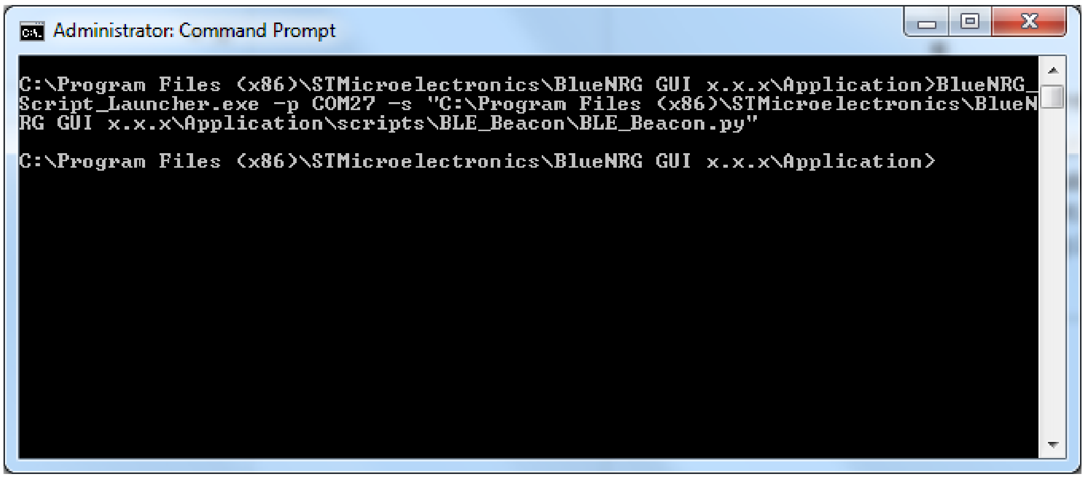
- COM27 (ST DK)

Options to run a script on a connected device

C:\Program Files (x86)\STMicroelectronics\BlueNRG GUI x.x.x\Application>BlueNRG_Script_Launcher.exe -p COM27 –s C:\Program Files (x86)\STMicroelectronics\BlueNRG GUI x.x.x\Application\scripts\BLE_Beacon \BLE_Beacon.py

COM27 is the platform virtual COM port.

Figure 21. **Script launcher: run a script**



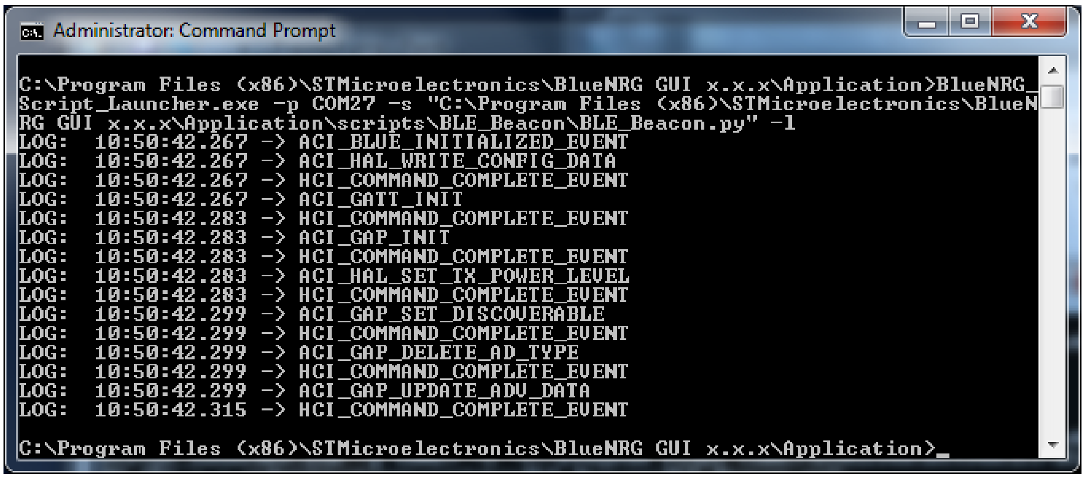
```
C:\Program Files (x86)\STMicroelectronics\BlueNRG GUI x.x.x\Application>BlueNRG_
Script_Launcher.exe -p COM27 -s "C:\Program Files (x86)\STMicroelectronics\BlueN
RG GUI x.x.x\Application\scripts\BLE_Beacon\BLE_Beacon.py"

C:\Program Files (x86)\STMicroelectronics\BlueNRG GUI x.x.x\Application>
```

Options to run a script on a connected device with log data

C:\Program Files (x86)\STMicroelectronics\BlueNRG GUI x.x.x\Application>BlueNRG_Script_Launcher.exe -p COM27 –s C:\Program Files (x86)\STMicroelectronics\BlueNRG GUI x.x.x\Application\scripts\BLE_Beacon \BLE_Beacon.py -l

COM27 is the platform virtual COM port.

Figure 22. **Script launcher: run a script with log data**



```
C:\Program Files (x86)\STMicroelectronics\BlueNRG GUI x.x.x\Application>BlueNRG_
Script_Launcher.exe -p COM27 -s "C:\Program Files (x86)\STMicroelectronics\BlueN
RG GUI x.x.x\Application\scripts\BLE_Beacon\BLE_Beacon.py" -l
LOG:  10:50:42.267 -> ACI_BLUE_INITIALIZED_EVENT
LOG:  10:50:42.267 -> ACI_HAL_WRITE_CONFIG_DATA
LOG:  10:50:42.267 -> HCI_COMMAND_COMPLETE_EVENT
LOG:  10:50:42.267 -> ACI_GATT_INIT
LOG:  10:50:42.283 -> HCI_COMMAND_COMPLETE_EVENT
LOG:  10:50:42.283 -> ACI_GAP_INIT
LOG:  10:50:42.283 -> HCI_COMMAND_COMPLETE_EVENT
LOG:  10:50:42.283 -> ACI_HAL_SET_TX_POWER_LEVEL
LOG:  10:50:42.283 -> HCI_COMMAND_COMPLETE_EVENT
LOG:  10:50:42.299 -> ACI_GAP_SET_DISCOVERABLE
LOG:  10:50:42.299 -> HCI_COMMAND_COMPLETE_EVENT
LOG:  10:50:42.299 -> ACI_GAP_DELETE_AD_TYPE
LOG:  10:50:42.299 -> HCI_COMMAND_COMPLETE_EVENT
LOG:  10:50:42.299 -> ACI_GAP_UPDATE_ADV_DATA
LOG:  10:50:42.315 -> HCI_COMMAND_COMPLETE_EVENT

C:\Program Files (x86)\STMicroelectronics\BlueNRG GUI x.x.x\Application>_
```

4 References

Table 8. Reference information

What	Where	Description
STSW-BNRGGUI	www.st.com/bluenrg-1, www.st.com/bluenrg-2, www.st.com/bluenrg-ms, tools and software section	BlueNRG GUI SW package
STSW-BLUENRG-DK	www.st.com/bluenrg-ms, evaluation tools section	BlueNRG DK SW package for BlueNRG, BlueNRG-MS kits
STSW-BLUENRG1-DK	www.st.com/bluenrg-1, www.st.com/bluenrg-2, evaluation tools section	BlueNRG-1 and BlueNRG-2 DK SW package for BlueNRG-1 and BlueNRG-2 BLE v2.x stack family

5 List of acronyms

Table 9. List of acronyms used in this document

Term	Meaning
BLE	Bluetooth low energy
DFU	Device firmware upgrade
HW	Hardware
IFR	Information register
SoC	System-on-chip
SW	Software
USB	Universal serial bus

Revision history

Table 10. Document revision history

Date	Version	Changes
24-May-2016	1	Initial release.
24-Jun-2016	2	Added reference to BlueNRG-1 device and minor text changes.
26-Jul-2016	3	Minor fixes for BlueNRG-1 support.
07-Oct-2016	4	Updated Section 2.1.1: "BlueNRG, BlueNRG-MS network coprocessors" and Section 2.2.2: "Tools"
20-Jun-2017	5	Minor text and formatting changes Added reference to BlueNRG-2 SoC
23-Oct-2017	6	Added: reference to Save As C Code utility, INSERT_INT_NUMBER and INSERT_FLOAT_NUMBER utility commands
05-Apr-2018	7	Updated Section 2.1.2 BlueNRG-1 and BlueNRG-2 network coprocessors, Section 2.2.1 GUI main window and Section 2.2.4 GUI script window.
07-Dec-2018	8	Updated Section 2.2.1 GUI main window, Section 2.2.3 GUI ACI utility window, Section 2.2.4 GUI script window.
05-Mar-2020	9	Updated figures throughout Section 2.2 The BlueNRG graphical user interface. Removed reference to the STSW-BNRG_V1DK in Section 4 References and added Section 2.2.7 GUI RF throughput window. Updated Section 2.1.2 BlueNRG-1 and BlueNRG-2 network coprocessors.

Contents

List of tables

List of figures