life.augmented

AN5471

Application note

The BlueNRG-LP UART bootloader protocol

Introduction

The BlueNRG-LP is a very low power Bluetooth low energy (BLE) single-mode system-on-chip, compliant with Bluetooth® specification. The architecture core is a "Cortex-M0+" 32-bit.

This application note contains the specifications of the BlueNRG-LP UART bootloader.

1 UART bootloader configuration

To communicate with the BlueNRG-LP bootloader the host UART has to be configured as follows:

- UART data 8-bit
- NO parity
- Stop bit 1
- NO flow control
- Baud rate range [500 – 460800]

The bootloader is configured to use the UART pin:

- UART RX = PA8
- UART TX = PA9

Note:

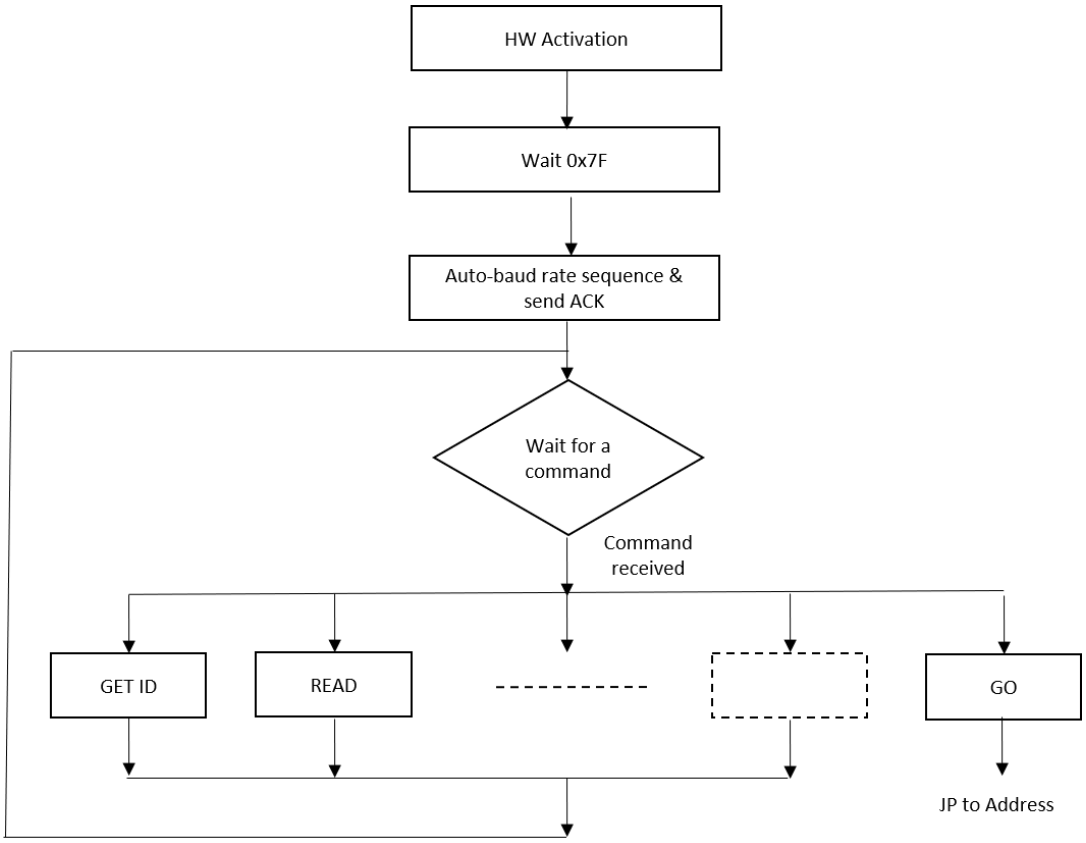
1. *Bootloader is available only on pins PA8, PA9*
2. *Bootloader is activated by hardware by forcing PA10 high during hardware reset (otherwise, the application residing in Flash is launched)*

2 UART bootloader activation

The BlueNRG-LP bootloader is activated by forcing PA10 high at device hardware reset. Once the bootloader is activated, the firmware initializes the BlueNRG-LP serial interface and starts a procedure to auto-detect the host UART baud rate and begins to scan the USART RX line pin, waiting for the 0x7F data frame from the host: one start bit, 0x7F data bits, no parity bit and one stop bit.

Once the baud rate is calculated an acknowledge byte (0x79) is returned to the host, which signals that the BlueNRG-LP is ready to receive commands.

Figure 1. UART bootloader for BlueNRG-LP devices



3 UART bootloader commands

The table below lists the supported commands, fully detailed in the following subsections.

Table 1. BlueNRG-LP UART bootloader commands

Command	Command code	Command description
Get List Command	0x00	Gets the version and the allowed commands supported by the current version of the bootloader
Get Version	0x01	Gets the bootloader version
Get ID	0x02	Gets the chip ID
Read Memory	0x11(1)	Reads up to 256 bytes of memory starting from an address specified by the application
Go	0x21(1)	Jumps to user application code located in the internal Flash memory or in RAM
Write Memory	0x31(1)	Writes up to 256 bytes to the RAM or Flash memory starting from an address specified by the application
Erase	0x43	Erases from one to all the Flash memory pages
Readout Protect	0x82	Enables the read protection
Readout Unprotect	0x92	Disables the read protection

1. *This command is disabled when readout protection is enabled.*

All communications from the host to the BlueNRG-LP device are safe because they are verified by:

- Checksum: received blocks of data bytes are XORed. A byte containing the computed XOR of all previous bytes is added to the end of each communication (checksum byte). By XORing all received bytes, data + checksum, the result at the end of the packet must be 0x00
- For each bootloader command, the host sends a byte and its complement
- Each packet is either accepted (ACK answer) or discarded (NACK answer):
  - ACK = 0x79
  - NACK = 0x1F

3.1 Get List command

The Get List command allows getting the version of the bootloader and the supported commands.

When the BlueNRG-LP bootloader receives the Get List command, it transmits the bootloader version and the supported command codes to the host.

Figure 2. Get List command: host side

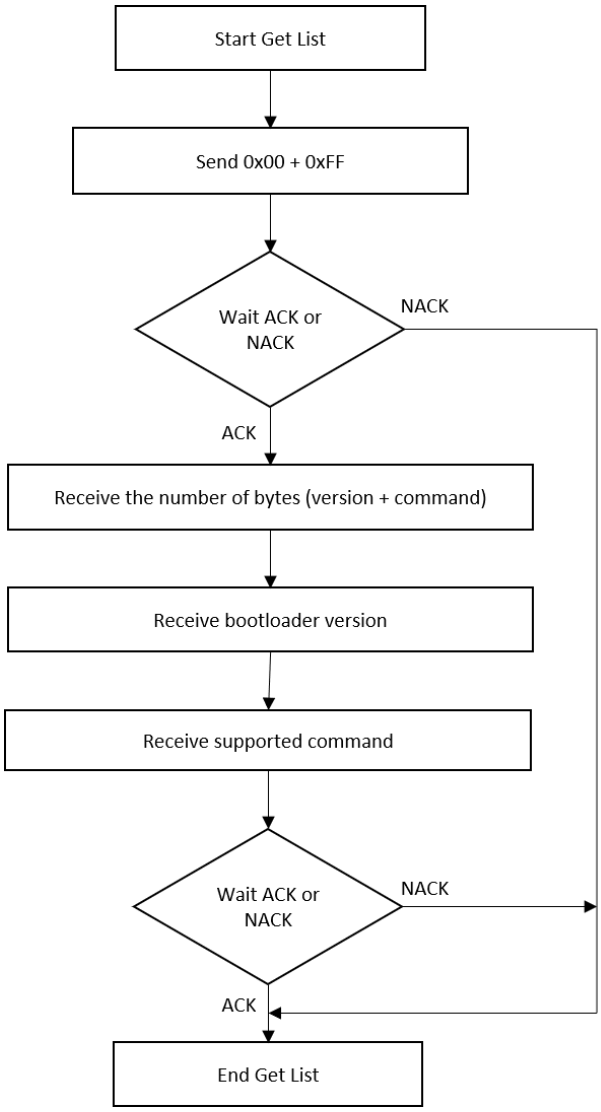
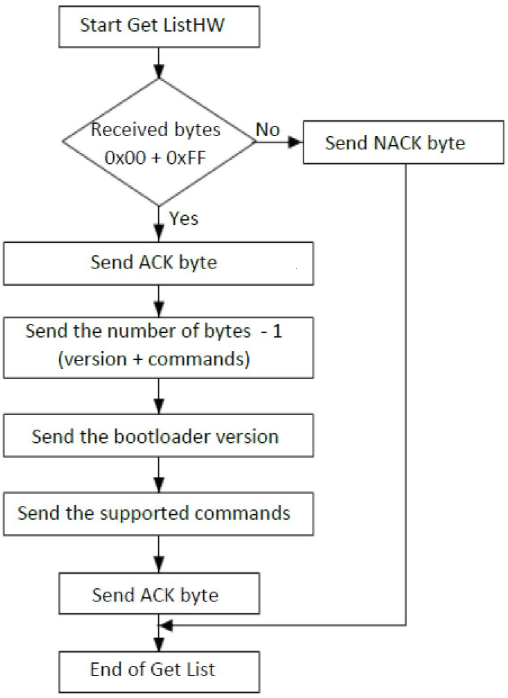


Figure 3. Get list command: device side



The BlueNRG-LP sends the bytes as follows:
Byte 1: ACK
Byte 2: N = 9 = the number of bytes to follow -1 except current and ACKs.
Byte 3: Bootloader version (0 < Version <= 255), example: 0x01 = Version 1.0
Byte 4: 0x00 – Get List command
Byte 5: 0x01 – Get Version command
Byte 6: 0x02 – Get ID command
Byte 7: 0x11 – Read Memory command
Byte 8: 0x21 – Go command
Byte 9: 0x31 – Write Memory command
Byte 10: 0x43 – Erase command
Byte 11: 0x82 – Readout Protect command
Byte 12: 0x92 – Readout Unprotect command
Byte 13: ACK. This is the last byte

3.2 Get Version command

The Get Version command is used to get the bootloader version. When the bootloader receives the command, it transmits the information shown below to the host.

Figure 4. Get Version command: host side

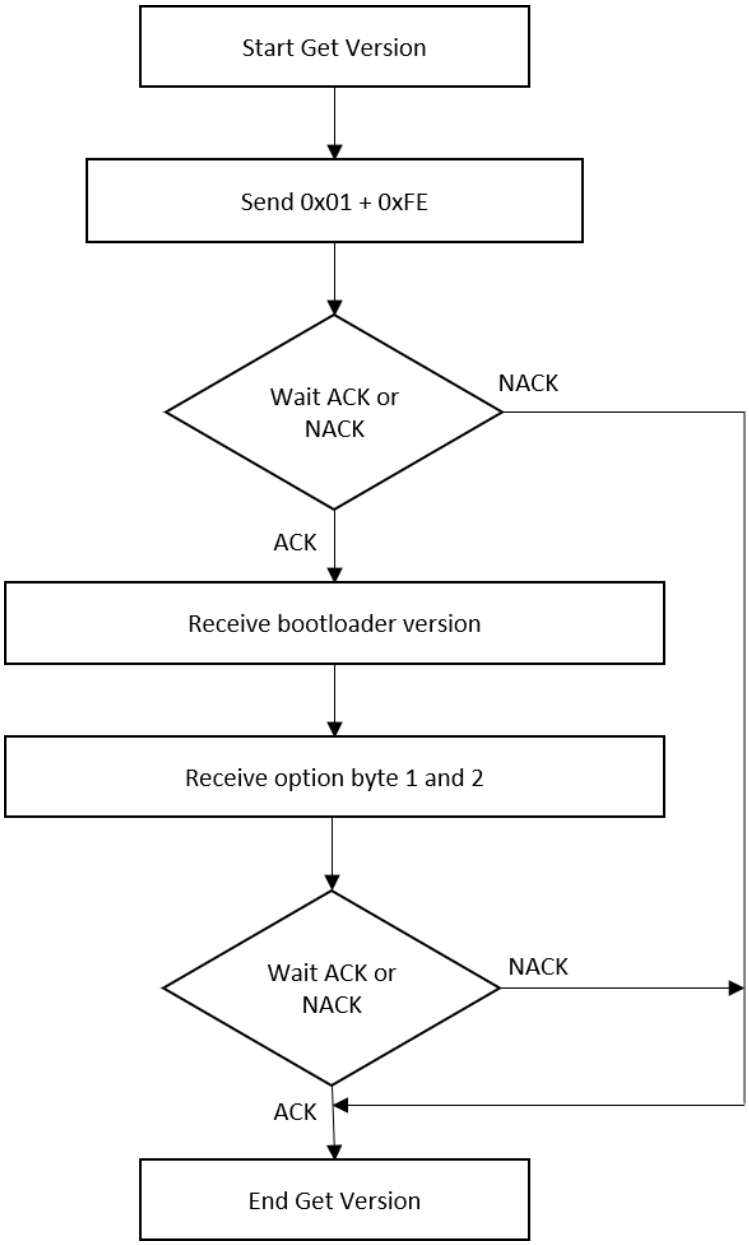
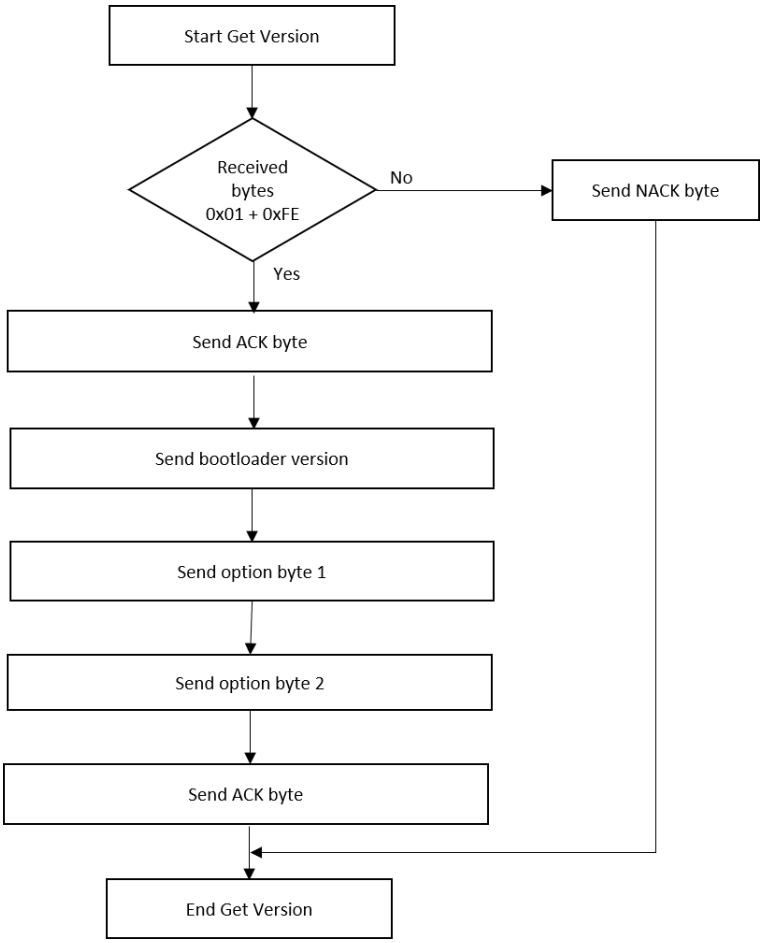


Figure 5. Get Version command: device side



The BlueNRG-LP device sends the bytes as follows:

Byte 1: ACK

Byte 2: Bootloader version (0 < Version <= 255), example: 0x01 = Version 1.0

Byte 3: Option byte 1: 0x00

Byte 4: Option byte 2: 0x00

Byte 5: ACK

3.3 Get ID command

The Get ID command is used to get the version of the chip ID (identification). When the bootloader receives the command, it transmits the product ID to the host.

Figure 6. **Get ID command: host side**

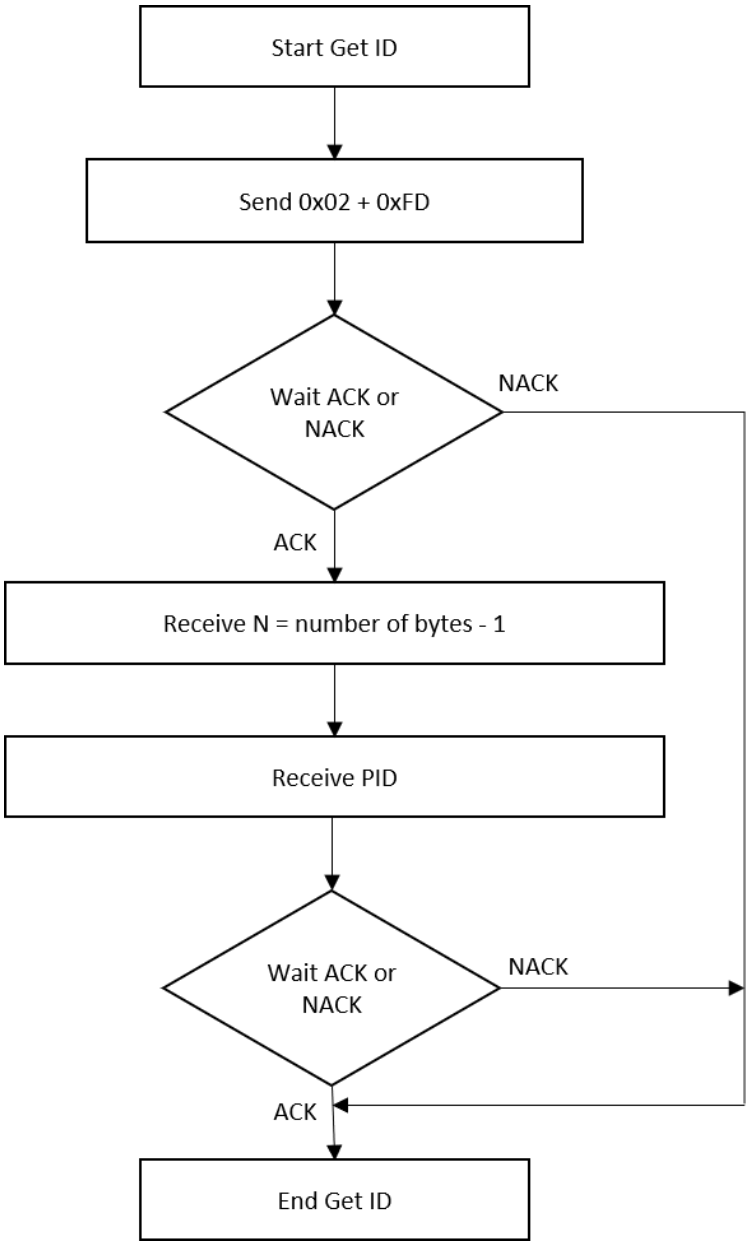
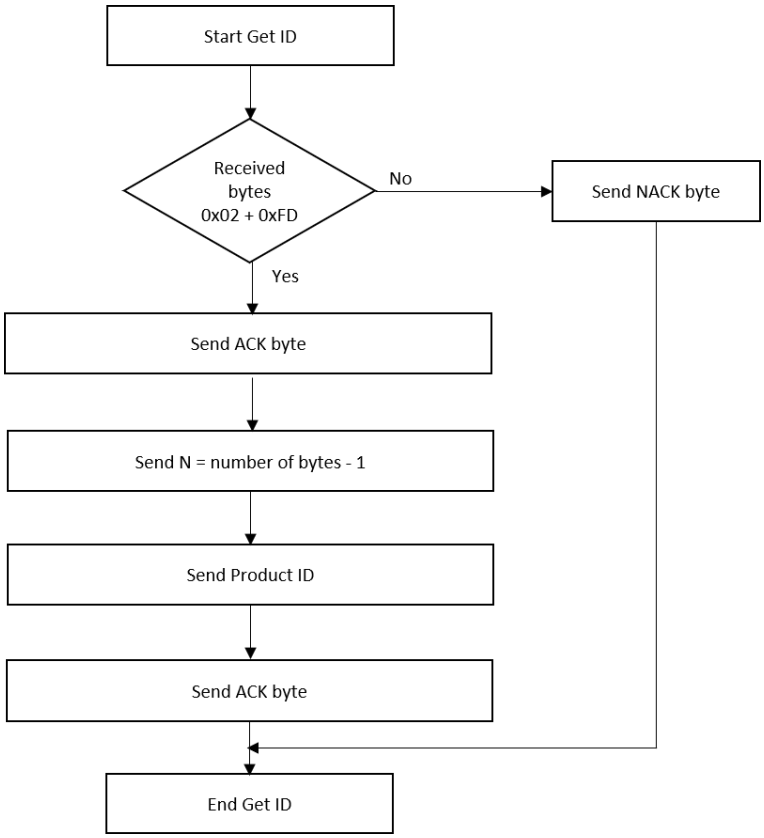


Figure 7. Get ID command: device side



The BlueNRG-LP device sends the bytes as follows:

Byte 1: ACK

Byte 2: 0x02: the number of bytes -1, except for current byte and ACKs

Byte 3-5: PID

Byte 6: ACK

Note: The Get ID command format returns three bytes:

- *BYTE1: Metal fix version (e.g. cut 1.0 is 0)*
- *BYTE2: Mask set version (e.g cut 1.0 is 1)*
- *BYTE3: is split into two nibbles: 0xHL:*
  - *H is the product ID:*
    - *3 means BlueNRG-LP*
  - *L is the flash size code:*
    - *F: 256 kB (for BlueNRG-LP)*

3.4 Read Memory command

The Read Memory command is used to read data from any valid memory address in RAM and Flash memory.

When the bootloader receives the Read Memory command, it transmits the ACK byte to the application. After which, the bootloader waits for an address (4 bytes, byte 1 is the address MSB and byte 4 is the LSB) and a checksum byte. If the received address is valid and the checksum is correct, the bootloader transmits an ACK byte, otherwise it transmits a NACK byte and aborts the command.

When the address is valid and the checksum is correct, the bootloader waits for N that is the number of bytes to be transmitted – 1 and for its complemented byte (checksum).

If the checksum is correct it transmits the needed data ((N + 1) bytes) to the application, starting from the received address.

If the checksum is not correct, it sends a NACK before aborting the command.
If the readout protection is active, a NACK byte is sent to the host when the Read Memory command is received.

Figure 8. Read Memory command: host side

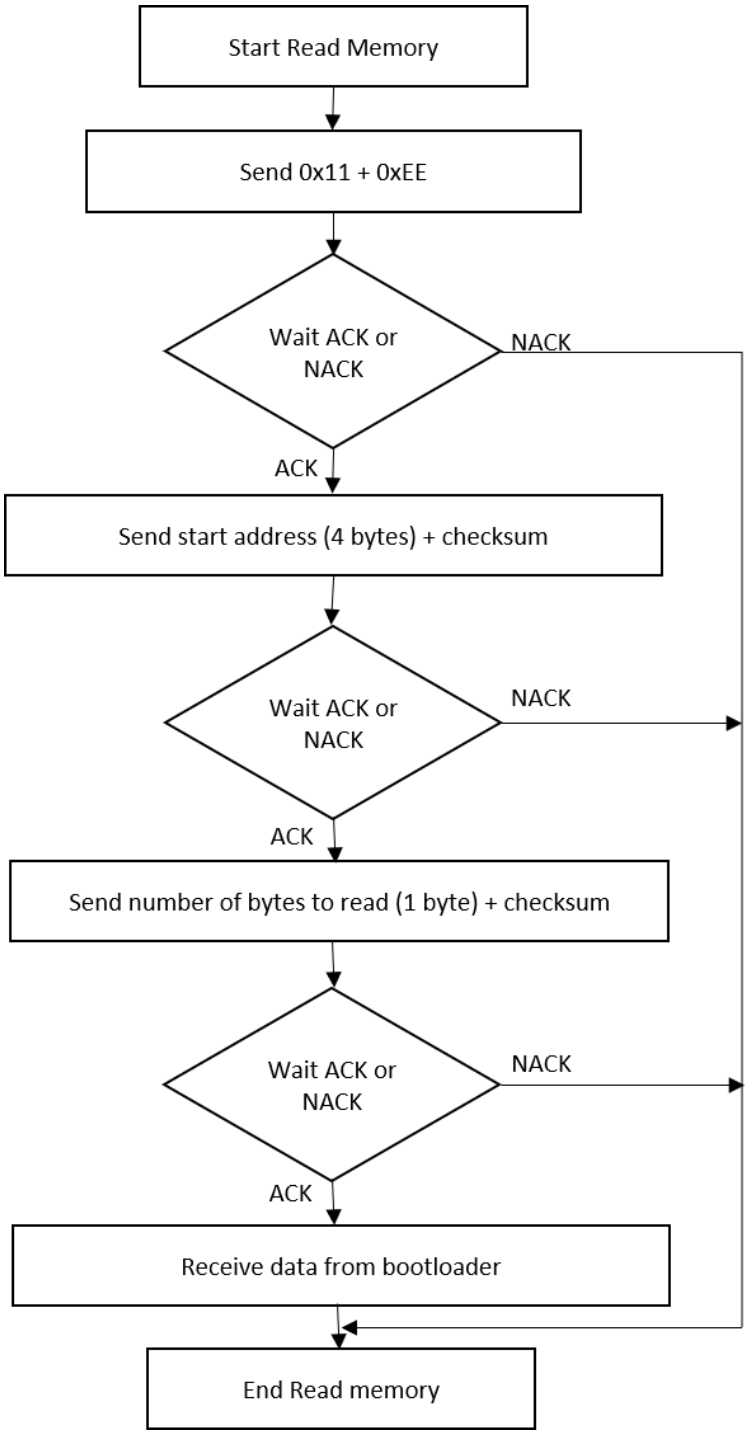
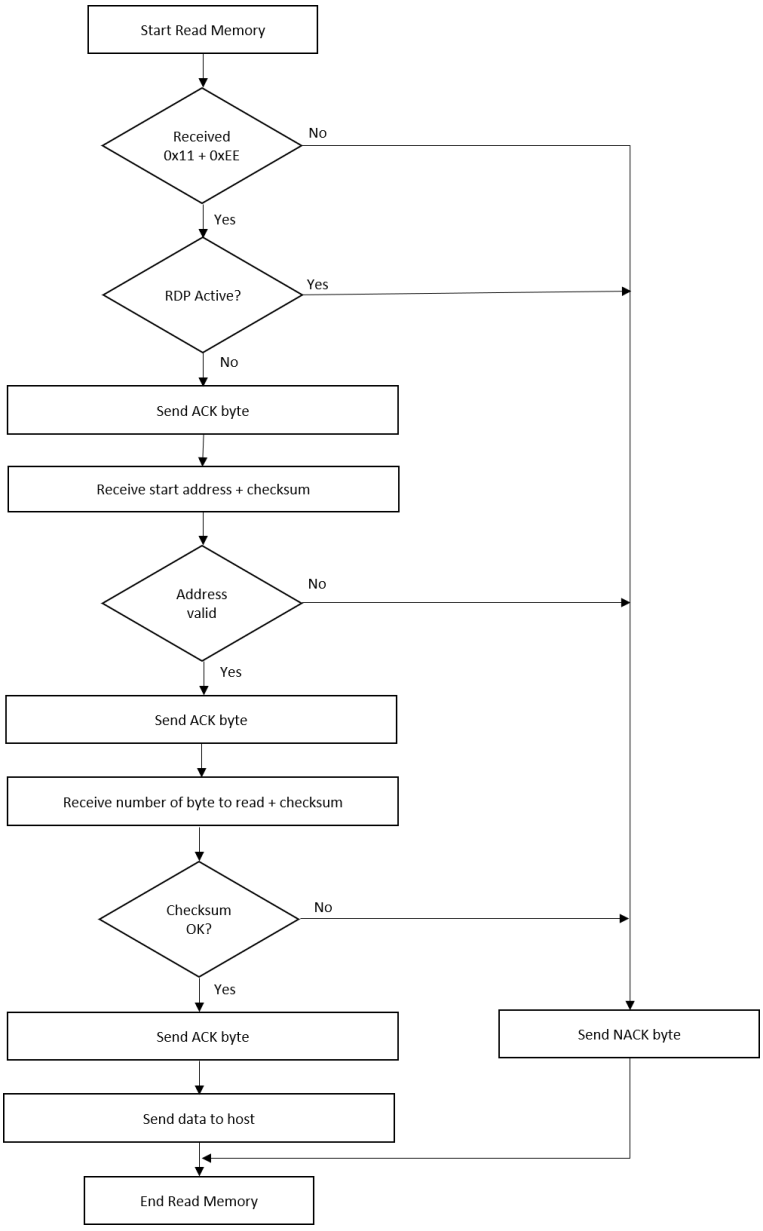


Figure 9. Read Memory command: device side



The host sends bytes to the BlueNRG-LP as follows:
Byte 1: 0x11
Byte 2: 0xEE
Wait for ACK
Byte 3 to 6: start address

- Byte 3: MSB
- Byte 6: LSB

Byte 7: Checksum: XOR of address bytes
Wait for ACK
Byte 8: Number of bytes to read -1 (0 < N <= 255)
Byte 9: Checksum: XOR byte 8 (complement of byte 8)

3.5 Go command

The Go command is used to execute the downloaded code or any other code by jumping to an address specified by the application. When the bootloader receives the Go command, it transmits the ACK byte to the application. After which, the bootloader waits for an address (4 bytes, byte 1 is the address MSB and byte 4 is LSB) and a checksum byte.

If the address is valid and the checksum is correct, the bootloader transmits an ACK byte, otherwise it transmits a NACK byte and aborts the command.

When the address is valid and the checksum is correct, the bootloader firmware performs the following actions:

- initializes the registers of the peripherals used by the bootloader to their default reset values
- initializes the user application main stack pointer
- jumps to the memory location programmed in the received ‘address + 4’ (which corresponds to the address of the application reset handler). For example, if the received address is 0x10040000, the bootloader jumps to the memory location programmed at address 0x10040004.

Figure 10. Go command: host side

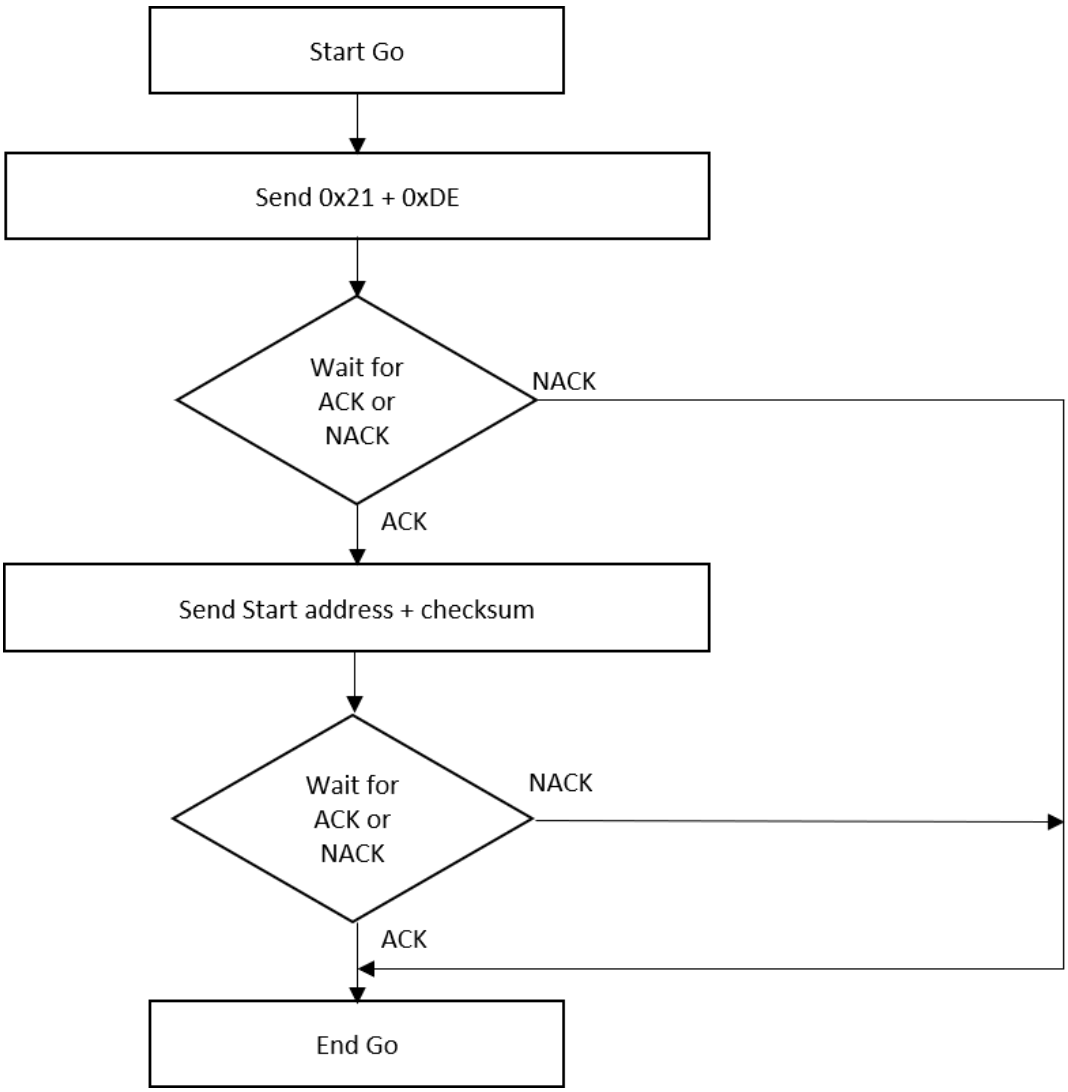
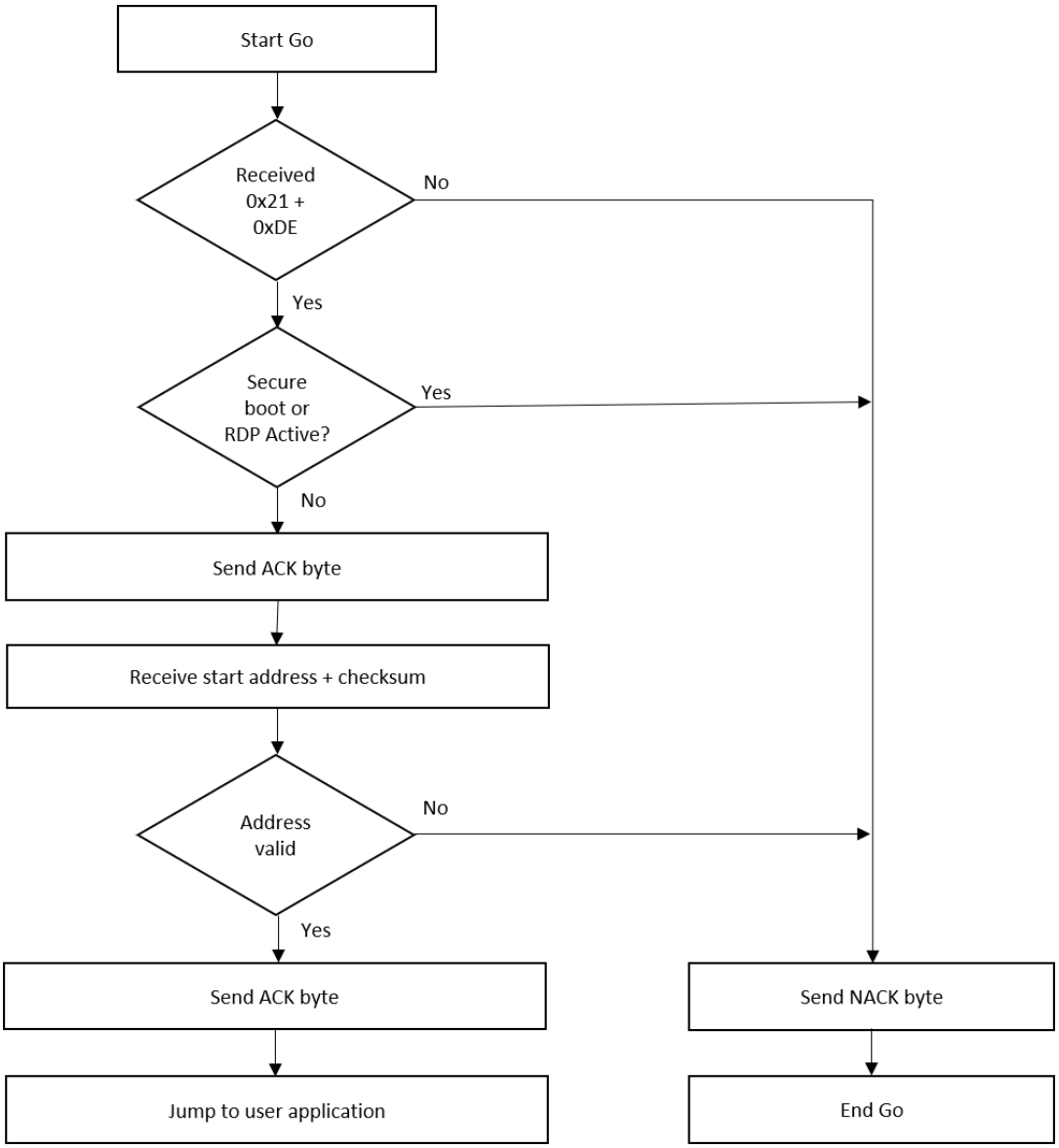


Figure 11. **Go command: device side**



The host sends bytes to the BlueNRG-LP as follows:
Byte 1: 0x21
Byte 2: 0xDE
Wait for ACK
Byte 3 – 6: start application address

- Byte 3: MSB
- Byte 6: LSB

Byte 7: checksum: XOR of address bytes
Wait for ACK

3.6 Write Memory command

The Write Memory command is used to write data to any valid memory address RAM or Flash memory.

When the bootloader receives the Write Memory command, it transmits the ACK byte to the application. After which, the bootloader waits for an address (4 bytes, byte 1 is the address MSB and byte 4 is the LSB) and a checksum byte.

If the received address is valid and the checksum is correct, the bootloader transmits an ACK byte, otherwise it transmits a NACK byte and aborts the command.

When the address is valid and the checksum is correct, the bootloader performs these actions:

- gets a byte, N, which contains the number of data bytes to be received
- receives the user data ((N + 1) bytes) and the checksum (XOR of N and of all data bytes)
- programs the user data to memory starting from the received address

At the end of the command, if the write operation is successful, the bootloader transmits the ACK byte; otherwise it transmits a NACK byte to the application and aborts the command.

The maximum length of the block to be written for the BlueNRG-LP is 256 bytes.

If the readout protection is active, a NACK byte is sent to the host when the Write Memory command is received.

Figure 12. **Write Memory command: host side**

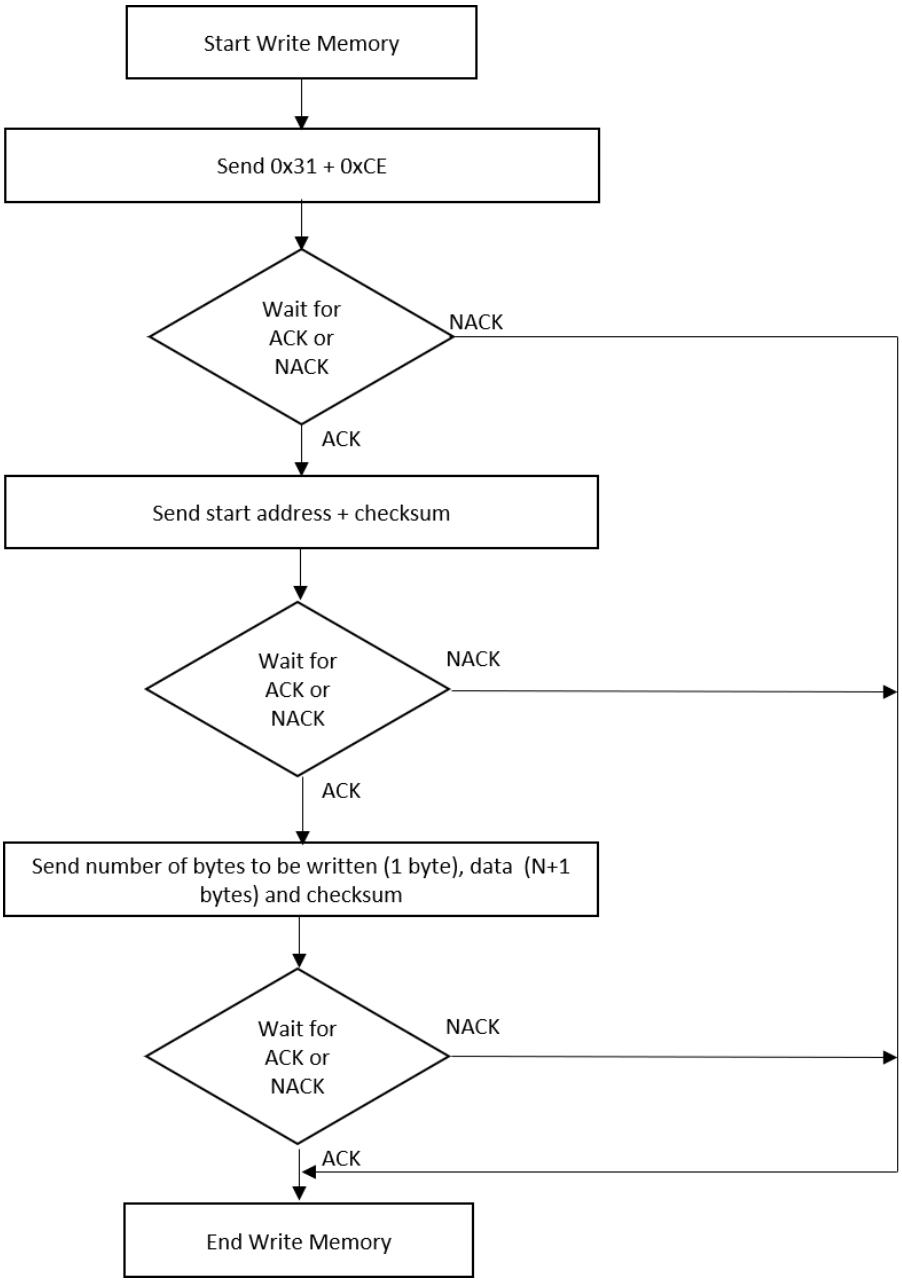
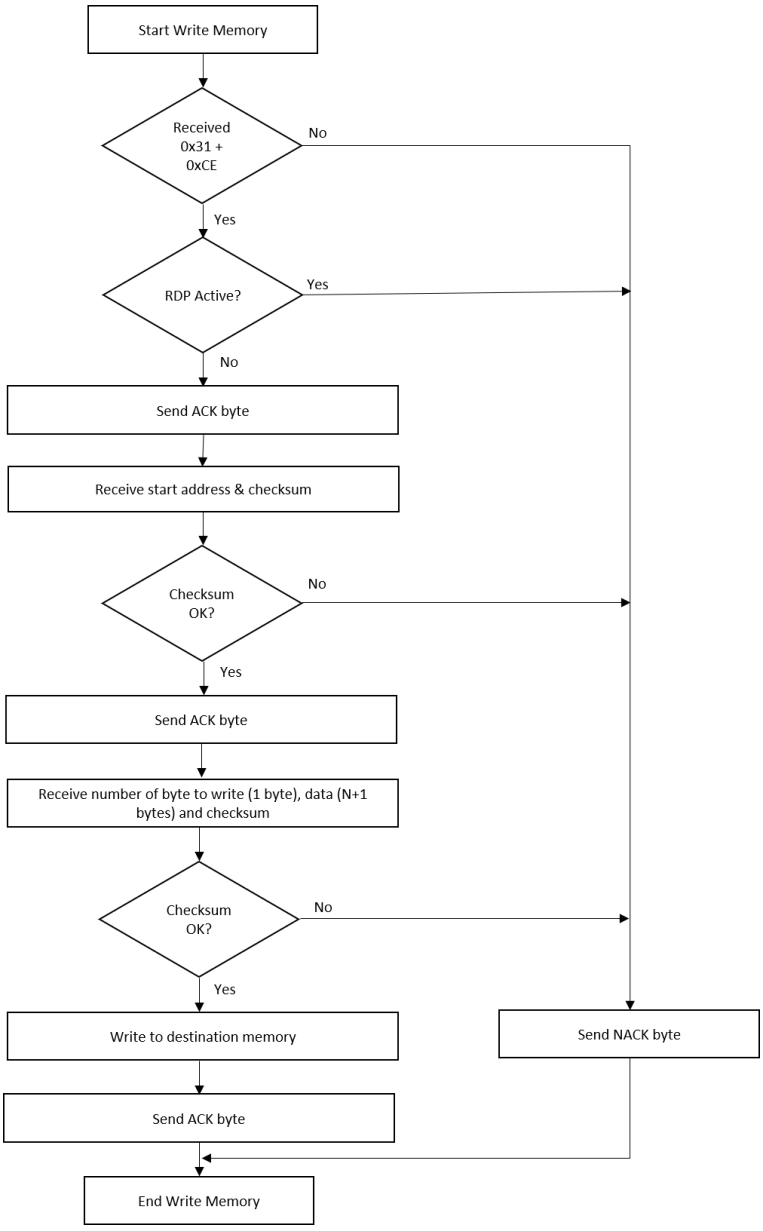


Figure 13. Write Memory command: device side



The host sends the bytes to the BlueNRG-LP as follows:
Byte 1: 0x31
Byte 2: 0xCE
Wait for ACK
Byte 3 to byte 6: start address

- Byte 3: MSB
- Byte 6: LSB

Byte 7: Checksum: XOR (byte 3, byte 4, byte 5, byte 6)
Wait for ACK
Byte 8: number of bytes to be received (0< N <= 255)
N + 1 data bytes: (Max. 256 bytes)
Checksum byte: XOR (N, N+1 data bytes)
Wait for ACK

3.7 Erase Memory command

The Erase Memory command allows the host to erase Flash memory pages. When the bootloader receives the Erase Memory command, it transmits the ACK byte to the host. After which, the bootloader receives one byte (number of pages to be erased), the Flash memory page codes and a checksum byte. If the checksum is correct, the bootloader then erases the memory and sends an ACK byte to the host, otherwise it sends a NACK byte to the host and the command is aborted.

Erase Memory command specifications are:

1. the bootloader receives one byte that contains N, the number of pages to be erased – 1. N= 255 is reserved for mass erase request. For 0 < N <= 79, N + 1 pages are erased.
2. the bootloader receives (N + 1) bytes, each byte containing a page number

Figure 14. Erase Memory command: host side

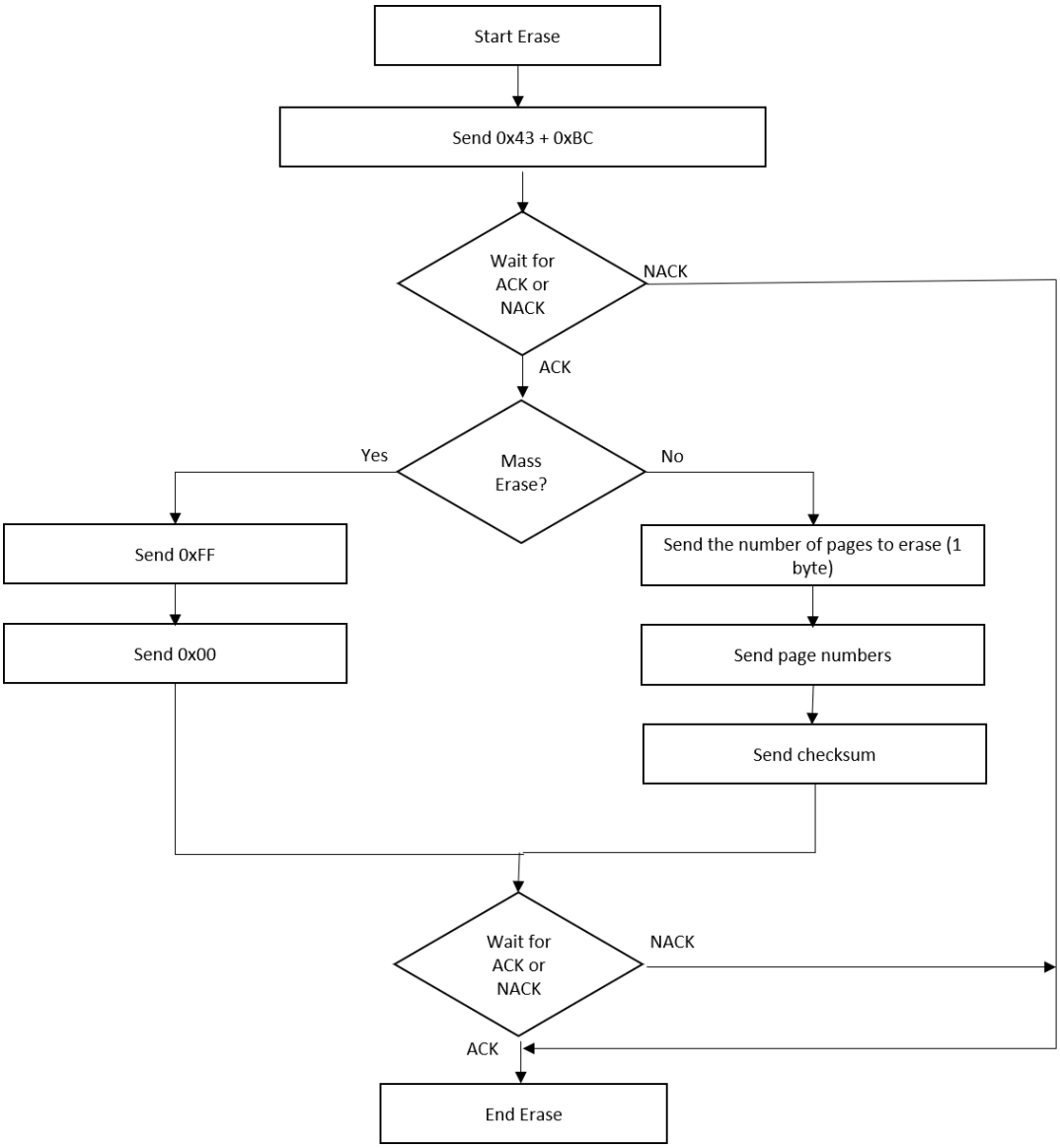
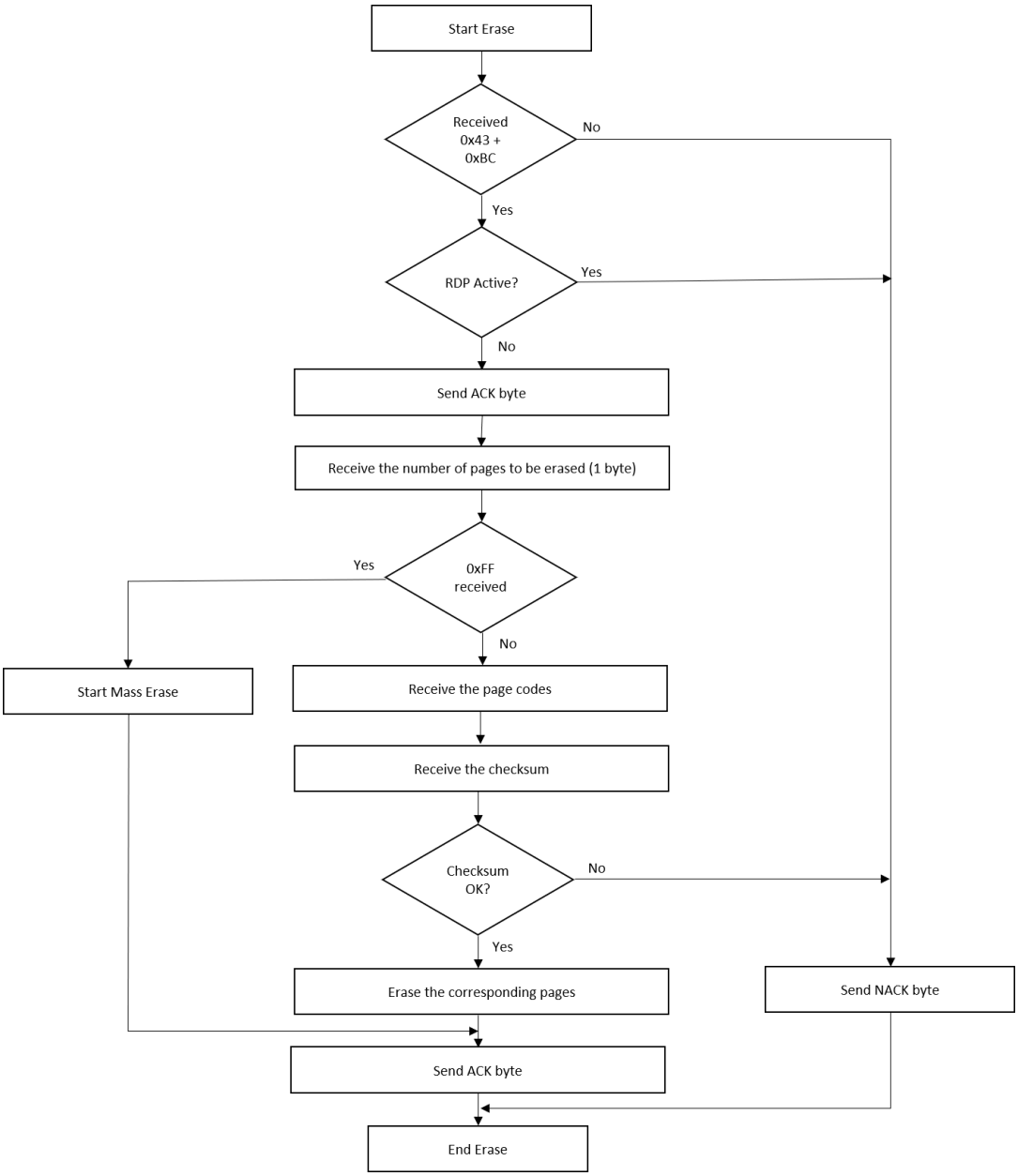


Figure 15. **Erase Memory command: device side**



The host sends bytes to the BlueNRG-LP as follows:

Byte 1: 0x43

Byte 2: 0xBC

Wait for ACK

Byte 3: 0xFF or number of pages to be erased – 1 (0 <= N <= maximum number of pages)

Byte 4: 0x00 (in case of mass erase) or ((N+1 bytes (page numbers) and then checksum XOR(N, N+1 bytes))

Wait for ACK

3.8 Readout Protect command

The Readout Protect command is used to enable the Flash memory read protection. When the bootloader receives the Readout Protect command, it transmits the ACK byte to the host. After which, the bootloader enables the read protection for the Flash memory.

At the end of the Readout Protect command, the bootloader transmits the ACK byte to signal the end of the command.

Figure 16. Readout Protect command: host side

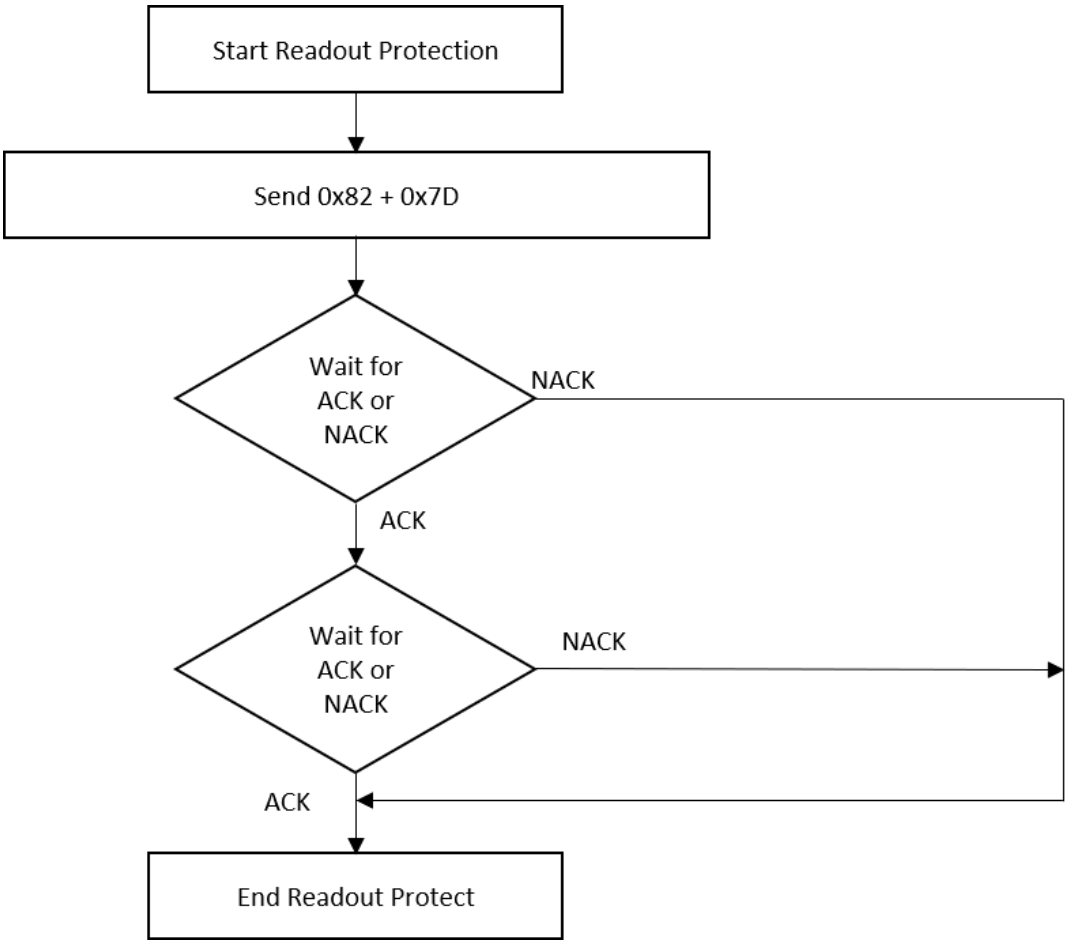
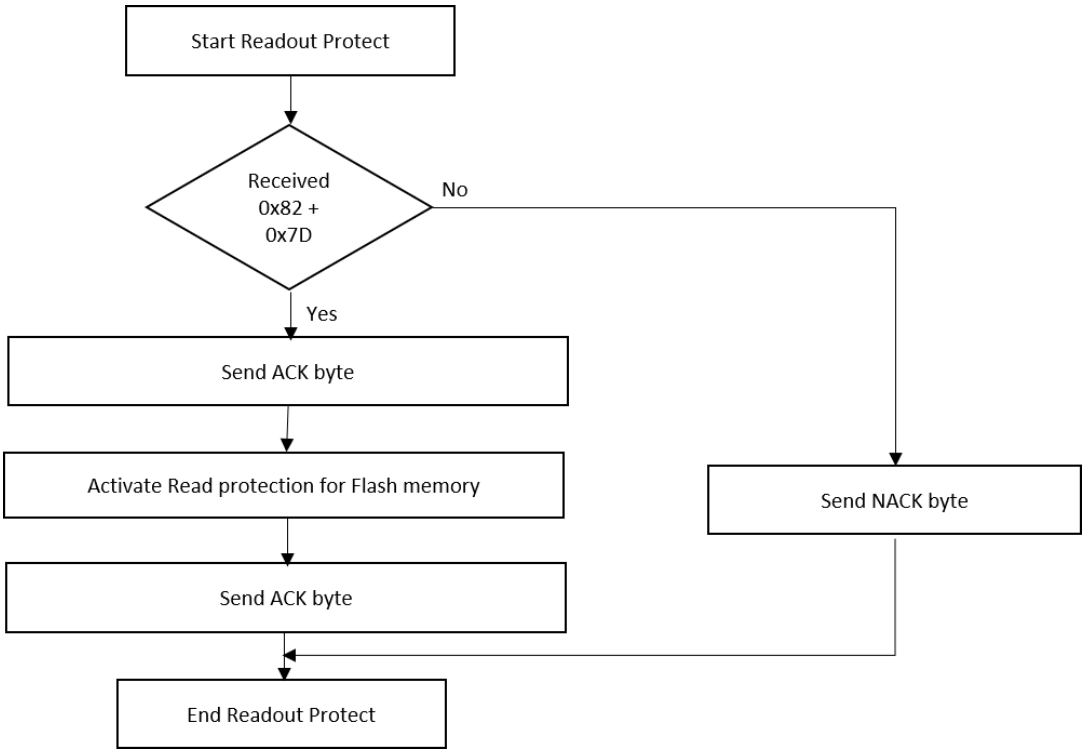


Figure 17. **Readout Protect command: device side**



3.9 Readout unprotect command

The Readout unprotect command is used to disable the Flash memory read protection. When the bootloader receives the Readout Unprotect command, it transmits the ACK byte to the host. After which, the bootloader erases all the Flash memory sectors and it disables the read protection for the entire Flash memory.

If the erase operation is successful, the bootloader deactivates the Readout Protection. If the erase operation is unsuccessful, the bootloader transmits a NACK and the read protection remains active.

At the end of the Readout Unprotect command, the bootloader transmits an ACK and generates a system Reset.

If other bootloader commands should be executed, the UART bootloader activation needs to be re-executed, that is HW reset plus PA10 high, as described in the UART bootloader activation.

Figure 18. **Readout unprotect command: host side**

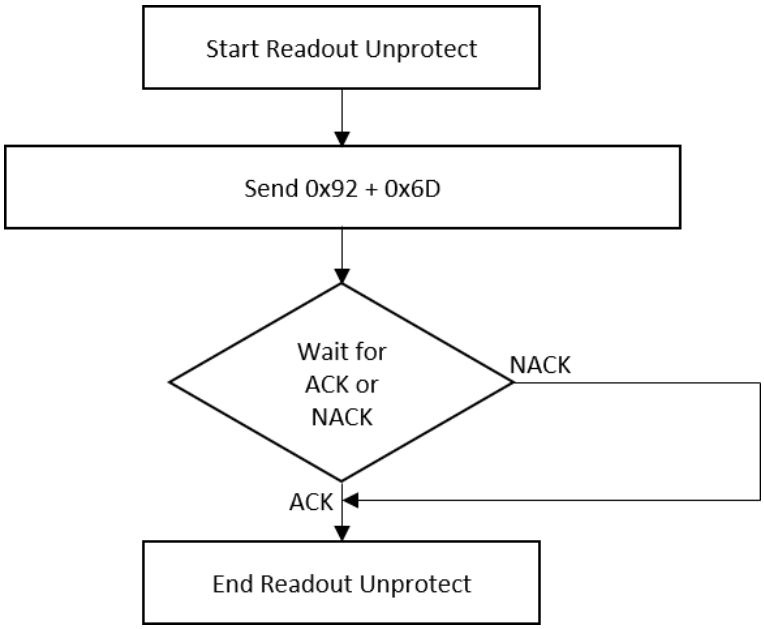
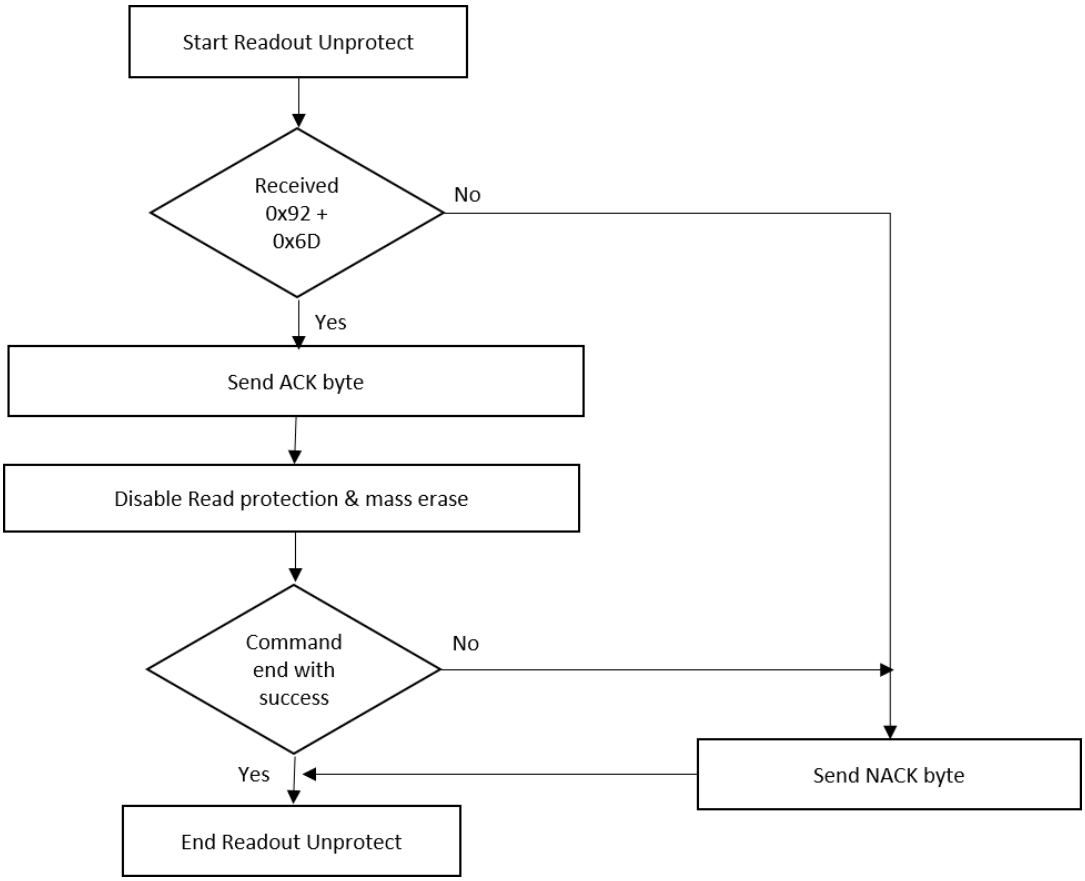


Figure 19. **Readout unprotect command: device side**



4 Secure bootloader

Secure bootloader feature is part of the UART bootloader preprogramed on the BlueNRG-LP at manufacturing time: it allows a user application to be authenticated in order to understand whether it is an authorized FW or not. Only authorized, trusted applications are allowed to run.

The BlueNRG-LP secure bootloader uses asymmetric cryptographic algorithms with key pair (public, private).

RSA-2048 is used with 256 bytes public key size.

This authentication method allows an Application FW to be authenticated by generating and appending a digital sign to it.

Secure bootloader feature can be activated by writing a specific value on OTP area.

4.1 How it works

The following steps describe the process to be followed in order to correctly authenticate an application FW at user production time:

User generates the key pair (public and private) and programs its device using signed FW:

- Application FW is signed by owner using the private key and the generated sign is appended to the application FW (digital sign size is 2048 bits)
- The BlueNRG-LP OTP section is used to store the generated public key

Secure bootloader uses the following components to verify/authenticate the application image:

- Stored public key on the BlueNRG-LP OTP
- Application FW image on Flash (with information related to the size)
- Digital signature appended at the end of the application FW image
- OTP dedicated location for enabling secure bootloader

Only the application FW image signed with the correct private key is executable and the private key is never shared or stored inside the BlueNRG-LP.

The key point of this method is that the application FW signing is done using the private key (not stored on the device) and authentication/verification is done with associated public key. So if the application FW has not been signed with the correct private key (owned by user), the BlueNRG-LP is not able to execute it.

4.1.1 How to store the public key in the OTP memory

The OTP memory is used to store the public key and other basic information needed to activate the secure bootloader feature.

To enable the secure bootloader feature, the OTP memory must contain the following information at specified addresses:

1. **Activation Word@ OTP address 0x10001800**
   - The activation word is: **0xEC1CC10B**
   - This word only is able to activate the secure boot feature in the bootloader code
2. **Start Flash Location of signed application FW @ OTP address 0x10001804**
   - This is the interrupt vector table start address of the signed application FW (i.e. 0x10040000)
3. **R2 Public Key @ OTP address 0x10001808**
   - R2 public key with length 256 bytes
4. **Modulus Public Key @ OTP address 0x10001908**
   - Modulus public key with length 256 bytes
5. **Exponent Public Key @ OTP address 0x10001A08**
   - Exponents public key with length 4 bytes

The total size of the information to store inside the OTP area is 524 bytes.

When all the data are stored inside the OTP area, it is **mandatory** to enable the OTP lock memory mechanism by writing a word different from 0xFFFFFFFF at OTP address 0x10001BFC.

4.1.2 How to sign an application FW

The following section provides a basic example of the following steps:

1. Generate public/private keys
2. Generate a digital sign from an input file
3. Append the digital sign to the input file
4. Store public key on the BlueNRG-LP OTP
5. Enable secure bootloader on BlueNRG-LP device

To enable the secure bootloader feature, the following secure bootloader utilities are released:

- **key_generation.exe:** this utility generates the public and private keys for the RSA-2048 algorithm, using the OPENSSL commands. The OPENSSL SW must be installed and the path must be defined within the Windows Path environment variable. This utility requires the following parameters:
  - -k <Destination folder for private and public key generated>
  - -f <Destination folder to save the public_key.c and public_key.py files>

  The option “k” creates a folder where the public and private key are saved.

  The option “f” creates a folder that contains the public key saved in two formats. The first format is compatible for C project and the second one for a python script.

  It is the choice of the user to create a simple C project that uses the “*public_key.c*” file to store the information in the OTP area. Alternatively, the “*public_key.py*” file is used from the “*store_key_OTP.exe*" utility.
- **create_signed_bin.exe:** this utility creates a signed FW starting from a binary file.

  This utility takes a not signed raw binary file in input and executes the following operations in the following order:
  1. Add the FW size inside the binary file at offset 0x18.
  2. Generate the signature for the application FW. To create the signature, the utility uses the private key already generated with the key_generation.exe utility.
  3. Add the signature to the end of the binary file and create a new signed file.

  This utility requires the following parameters:
  - -k <Folder with private and public key used to create the signature (the same folder created with the option “k” of the key_generation.exe utility)>
  - -f <Input raw binary image file not signed>
  - -o <Output raw binary image file signed>

  The file generated with the option “o” is the application signed that can be stored in the main FLASH.
- **store_key_OTP.exe**: this utility stores all the information inside the OTP area. It uses the bootloader engine, so, when this utility runs the device must be in bootloader mode (see Section 2 UART bootloader activation).

  The utility requires the following parameters:
  - -f <Folder with the public_key.py file to store in OTP (the same folder created with the option “f” of the key_generation.exe utility)>
  - -p <Device COM Port, used to run the bootloader utility>
  - -a <Start FW address (i.e. 0x10040000)>
  - -l <Lock of the OTP area>

Note: The secure bootloader utilities are available on the STSW-BNRGLP-DK, Utility, Secure_bootloader folder.

The secure boot feature is not reversible and cannot be disabled. When the secure boot feature is activated at reset, the device verifies whether the FW, in the main FLASH, is authenticated or not. The public key stored inside the OTP is used. The OTP cannot be changed, so, it is important to save the keys generated with the utility “key_generation.exe”, because if a new FW update is required, the same keys must be used to generate the signature for the new signed FW, otherwise the secure bootloader detects the FW as not authenticated and it is not executed.

Note: The -a Start FW address allows the authentication procedure to be applied to an image to be downloaded at a specific address. This capability could be useful in a scenario as OTA Service Manager, where the downloaded application, at a specific Flash location, could be authenticated using the secure bootloader mechanism.

If a new application loaded through OTA Service Manager is successfully authenticated, the secure bootloader allows OTA Service Manager to be jumped to and then give control to the authenticated application. Otherwise, no application is executed.

Revision history

Table 2. **Document revision history**

Date	Version	Changes
20-Jul-2020	1	Initial release.
29-Sep-2020	2	Update Section Introduction, Section 3.1 Get List command, Section 4.1.2 How to sign an application FW.

Contents

List of figures