AN4491
Application note

BlueNRG, BlueNRG-MS updater

Introduction

This document describes the updater functionality of BlueNRG and BlueNRG-MS devices.

Note: The document content is valid for both BlueNRG and BlueNRG-MS devices. Any specific difference is highlighted whenever it is needed.

Contents

1 Description

The updater is an independent program that resides within the BlueNRG device. It allows changing of the BlueNRG firmware, by receiving a new version via serial port (SPI), and reprogramming the Flash memory. The updater cannot change itself.

The updater is allocated in a section of the NVM (non-volatile-memory) of the device (Flash or ROM). When the updater is in Flash, the first sector is used, i.e. 2 KB.

Figure 1 and *Figure 2* below show the Flash layouts of the BlueNRG and BlueNRG-MS devices. Specific software such as the IFR, and the BlueNRG and BlueNRG-MS firmware stacks are indicated in addition to the area where the Updater is stored. On BlueNRG-MS Updater is stored on device ROM.

Figure 1. BlueNRG Flash layout

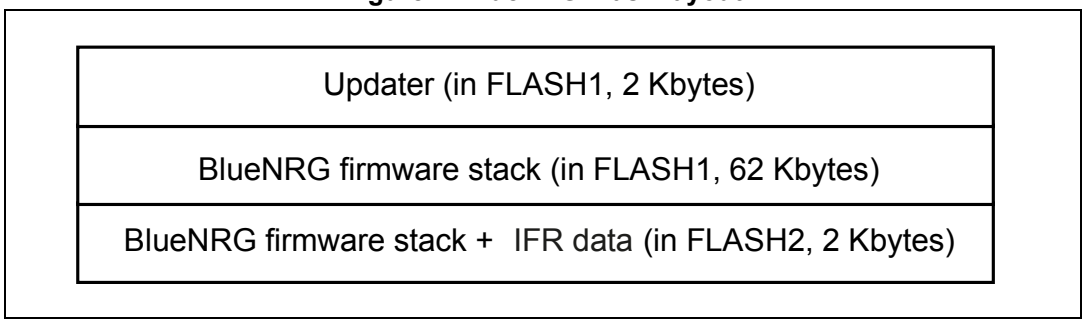
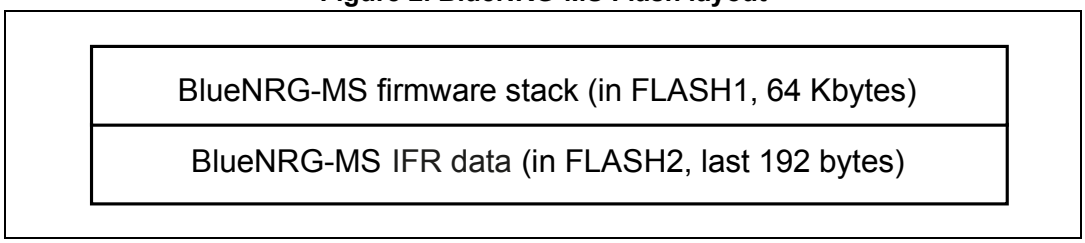


Figure 2. BlueNRG-MS Flash layout



On BlueNRG, as a separate program, the updater is able to work without the BlueNRG firmware. Even if the BlueNRG firmware is corrupted, or if the 62 KB firmware region and/or 2 KB IFR Flash are empty, the updater is still able to turn on the system, wait to receive new firmware, and then update the Flash. So it can safeguard the system and bring it back to a known working state.

2 CRC

The updater can perform 32-bit CRC calculation to verify the data stored inside the Flash. This is done by an updater command. The command returns the CRC result and the user can compare it with the previously calculated values. If it does not match, it means the Flash data is corrupted.

The common IEEE CRC 32 polynomial is used:

$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$

The user must choose at least 1 Flash sector (2 KB) to do the CRC calculation. It is also allowed to do CRC calculation for more continuous FLASH sectors.

3 Updater BLUE flag

There are 4 bytes of data stored in the firmware region as a flag (called BLUE flag). When the flag is set to a particular known value, it indicates that there is a valid firmware stored in the Flash. When new firmware programming is started, a special command should first be called to erase the BLUE flag. The flag should be set only at the end of the programming phase, if the firmware is correctly updated. So, if the firmware update procedure is not properly done, the flag is not set and the updater will be forced to stay in the program mode. This mechanism prevents BlueNRG from running a corrupted firmware that will cause unexpected malfunctioning.

Note: The BlueFlag word is stored at the 0x100108C0 location.

4 Entering updater mode

The updater can be started in two different ways:

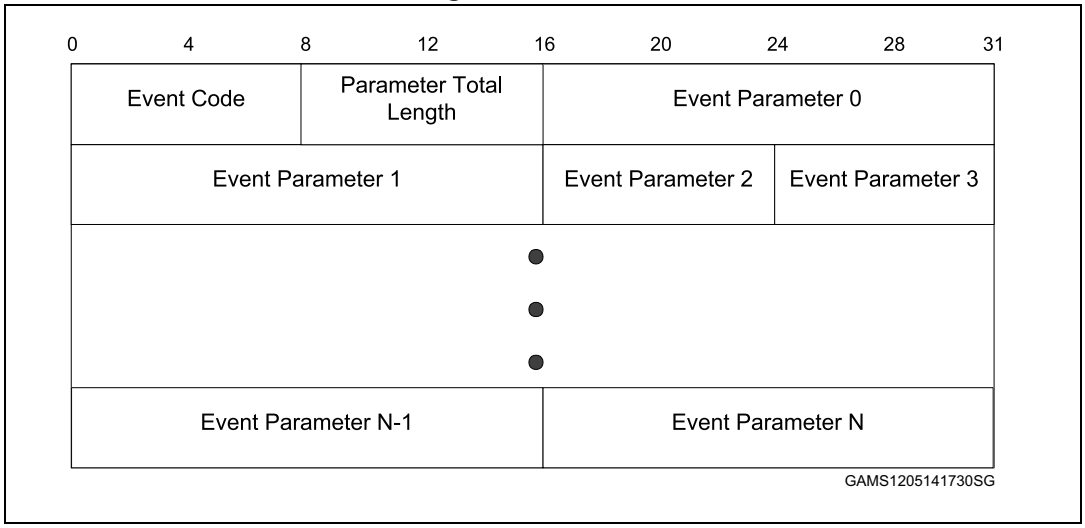
- Through the ACI command, when in normal operation (see *Section 6.1: aci_updater_start*).
- Using the IRQ pin during hardware reset. If the IRQ pin is detected high during BlueNRG startup (power-up or hardware reset), BlueNRG enters updater mode. The startup phase of BlueNRG (hardware and firmware) takes about 3 ms. To be sure that IRQ is detected high during startup, it is recommended to keep it high for at least 4 ms after Reset has been released.

5 Evt_BlueBlue_Initialized

When the BlueNRG firmware is started normally, it gives a Evt_Blue_Initialized event to the user to indicate the system has started.

The Evt_Blue_Initialized event is an ACI event with the same format as the other events.

Figure 3. ACI event



It is a vendor specific HCI event (event code 0xFF). In *Table 1* all the fields are described.

Table 1. Evt_Blue_Initialized event

Parameter	Size	Description
BlueNRG Event Code	1 byte	0x0001 - The event code for Evt_Blue_Initialized event
Reason code	1 byte	0x01 – Application started properly 0x02 – Updater mode entered because of Updater_Start command 0x03 - Updater mode entered because of a bad BLUE flag 0x04 - Updater mode entered because of IRQ pin high

6 Updater commands

Once updater mode is entered, the updater first reports a Evt_Blue_Initialized event, as described in the previous section. Then it continues checking the SPI interface to receive commands. The updater only responds to certain updater commands, listed below. The updater supports no other commands. Once entered, the updater stays in this mode unless clearly asked to exit by a command (aci_updater_reboot) or a HW reset.

For each command, the updater acknowledges with standard Command Complete event. The commands and events are in the format as the same as other ACI commands.

The updater supports only the following commands:

- aci_updater_start
- aci_updater_reboot
- aci_get_updater_version
- aci_get_updater_buffer_size
- aci_erase_blue_flag
- aci_reset_blue_flag
- aci_updater_erase_sector
- aci_updater_program_data_block
- aci_updater_read_data_block
- aci_updater_calc_crc
- aci_updater_hw_version

6.1 aci_updater_start

Table 2. aci_updater_start

Command name	Parameters	Return
aci_updater_start (0xFC20)		Status

Description:

This command is only implemented together with the normal application. The updater does not support this command. If this command is called, the system reboots and enters updater mode.

Table 3. aci_updater_start return parameters

Parameter	Size	Description
Status	1 byte	0x00: BLE_STATUS_SUCCESS

Event(s) generated:

The controller will generate a command complete event.

6.2 aci_updater_reboot

Table 4. aci_updater_reboot

Command name	Parameters	Return
aci_updater_reboot (0xFC21)		Status

Description:

This command reboots the system. This command does not set the BLUE flag, which must be done by another command.

Table 5. aci_updater_reboot return parameters

Parameter	Size	Description
Status	1 byte	0x00: BLE_STATUS_SUCCESS

Event(s) generated:

The controller will generate a command complete event.

6.3 aci_get_updater_version

Table 6. aci_get_updater_version

Command name	Parameters	Return
aci_get_updater_version (0xFC22)		Status Version

Description:

This command returns the version of the Updater.

Table 7. aci_get_updater_version return parameters

Parameter	Size	Description
Status	1 byte	0x00: BLE_STATUS_SUCCESS
Version	1 byte	0x00: If an error occurred 0xXX: Updater Version ID

Event(s) generated:

The controller will generate a command complete event.

6.4 aci_get_updater_buffer_size

Table 8. aci_get_updater_buffer_size

Command name	Parameters	Return
aci_get_updater_buffer_size (0xFC23)		Status Buffer Size

Description:

Return the maximum buffer size. This value limits the size of the data blocks that could be used on the command aci_updater_program_data_block.

Table 9. aci_get_updater_buffer_size return parameters

Parameter	Size	Description
Status	1 byte	0x00: BLE_STATUS_SUCCESS
Buffer Size	2 bytes	Size of the buffer, unit in bytes

Event(s) generated:

The controller will generate a command complete event.

6.5 aci_erase_blue_flag

Table 10. aci_erase_blue_flag

Command name	Parameters	Return
aci_erase_blue_flag (0xFC24)		Status

Description:

This command erases the BLUE flag in the Flash. After this operation, the updater cannot jump to the firmware until the BLUE flag is set to a valid value with aci_reset_blue_flag.

This command is strongly recommended when the updater wants to upgrade the firmware application.

Table 11. aci_erase_blue_flag return parameters

Parameter	Size	Description
Status	1 byte	0x00: BLE_STATUS_SUCCESS

Event(s) Generated:

The controller will generate a command complete event.

6.6 aci_reset_blue_flag

Table 12. aci_reset_blue_flag

Command name	Parameter	Return
aci_reset_blue_flag (0xFC25)		Status

Description:

Reset the BLUE flag to its proper value. This command must be called when the firmware upgrade is finished. So that after reboot, the update may jump to the firmware application.

Table 13. aci_reset_blue_flag return parameters

Parameter	Size	Description
Status	1 byte	0x00: BLE_STATUS_SUCCESS 0x4A: Blue Flag failed to be set The reason of failure is normally because the Flash sector is not erased before using the ResetBlueFlag command.

Event(s) generated:

The controller will generate a command complete event.

6.7 aci_updater_erase_sector

Table 14. aci_updater_erase_sector

Command name	Parameters	Return
aci_updater_erase_sector (0xFC26)	Sector Base Address	Status

Description:

This command erases one sector of the Flash memory. One sector is 2 KB. After erasing, the sector will be all 0xFF.

Table 15. aci_updater_erase_sector command parameters

Parameter	Size	Description
Address	4 bytes	Base address of the sector to be erased Sector 0: 0x10010000 (Updater sector, forbidden) Sector 1: 0x10010800 Sector 2: 0x10011000 Sector 3: 0x10011800 Sector 4: 0x10012000 Sector 5: 0x10012800 ... Sector 30: 0x1001F000 Sector 31: 0x1001F800 Sector 32: 0x10020000 (IFR sector) Only the based address listed here from Sector 1 - 32 are allowed. Other values will be rejected

Note: Don't erase sector 0.

Table 16. aci_updater_erase_sector return parameters

Parameter	Size	Description
Status	1 byte	0x00: BLE_STATUS_SUCCESS 0x12: Invalid HCI parameter If the base address given in the command is not the one listed.

Event(s) generated:

The controller will generate a command complete event.

6.8 aci_updater_program_data_block

Table 17. aci_updater_program_data_block

Command name	Parameters	Return
aci_updater_program_data_block (0xFC27)	Base Address Data Length Data	Status

Description:

This command writes a block of data to the Flash, starting from the given base address.

Table 18. aci_updater_program_data_block command parameters

Parameter	Size	Description
Address	4 bytes	Initial address to start the writing of the data
Data Length	2 bytes	Length of the data (unit in bytes)
Data	0-N bytes	Data to be programmed

Table 19. aci_updater_program_data_block return parameters

Parameter	Size	Description
Status	1 byte	0x00: BLE_STATUS_SUCCESS 0x12: Invalid HCI parameters This may happen if there is an attempt to write outside the allowed range.

Event(s) generated:

The controller will generate a command complete event went the data has been programmed.

6.9 aci_updater_read_data_block

Table 20. aci_updater_read_data_block

Command name	Parameters	Return
aci_updater_read_data_block (0xFC28)	Base Address Data Length	Status Data

Description:

This command reads a block of data from the Flash, starting from the given base address.

Note: For it is only allowed to read from the IFR flash. So the Base Address must be bigger than 0x10020000.

Table 21. aci_updater_read_data_block command parameters

Parameter	Size	Description
Address	4 bytes	Initial address
Data	2 bytes	Length of the data (unit in bytes)

Table 22. aci_updater_read_data_block return parameters

Parameter	Size	Description
Status	1 byte	0x00: BLE_STATUS_SUCCESS 0x12: Invalid HCI parameters This may happen if there is an attempt to read outside the allowed range
Data	0-N bytes	Data read

Event(s) generated:

The controller will generate a command complete event.

6.10 aci_updater_calc_crc

Table 23. aci_updater_calc_crc

Command name	Parameters	Return
aci_updater_calc_crc (0xFC29)	Base Address Number of Sectors	Status CRC Value

Description:

It calculates the CRC32 of one or more Flash sectors. One Flash sector is 2 KB (see also *Section 2*).

Table 24. aci_updater_calc_crc command parameters

Parameter	Size	Description
Address	4 bytes	Base address of the sector to calculate CRC Sector 0: 0x10010000 Sector 1: 0x10010800 Sector 2: 0x10011000 Sector 3: 0x10011800 Sector 4: 0x10012000 Sector 5: 0x10012800 ... Sector 30: 0x1001F000 Sector 31: 0x1001F800 Sector 32: 0x10020000 (IFR sector) Only the values listed here are allowed. Other values will be rejected.
Number of sectors	1 byte	Number of continuous Flash sectors to be calculated. Should be between 1 and 32. E.g. if the base address is 0x10011000, and the number of sectors is 3, it will calculate the 6 KB address starting from 0x10011000

Table 25. aci_updater_calc_crc return parameters

Parameter	Size	Description
Status	1 byte	0x00: BLE_STATUS_SUCCESS 0x12: Invalid HCI parameters This may happen if there is an attempt to read outside the allowed range
CRC Value	4 bytes	CRC32 value calculated for the region

Event(s) generated:

The controller will generate a command complete event.

7 Firmware update algorithm

The updater client should adhere to the following procedure when updating the firmware stack:

1. Enter updater mode with IRQ pin or aci_updater_start command
2. Wait for the Evt_Blue_Initialized and check the reason code to be sure that the device has booted in updater mode
3. Read updater version (optional).
4. Read updater buffer size to know the maximum data block that can be written in a single aci_updater_program_data_block command.
5. Clear BLUE flag.
6. Erase sectors containing the firmware:
   a) Sectors from 1 to 32 on BlueNRG (sector 0 contains bootloader and it is forbidden to erase or write into it). Sector 32 must be erased since it also contains firmware code.
   b) Sectors from 0 to 31 on BlueNRG-MS. Erase also sector 32 if IFR configuration is programmed during firmware update phase.
7. Write new firmware data with aci_updater_program_data_block command and IFR configuration if sector 32 has been erased.
8. Run aci_updater_calc_crc command, which generates the CRC calculation based on the loaded firmware.
9. If the CRC is OK, reset the BLUE flag. Otherwise notify the application or retry the whole procedure.
10. Reset the device or send aci_updater_reboot command to reboot the system. If IFR is changed during the update, a hardware reset is mandatory. It is also possible to switch off the power.

Note: This algorithm assumes that the firmware image has the BLUE flag at offset 0x08C0 set to 0xFFFFFFFF.

8 IFR update algorithm

The updater client should adhere to the following procedure when updating the IFR configuration:

1. Enter updater mode with IRQ pin or aci_updater_start command
2. Wait for the Evt_Blue_Initialized and check the reason code to be sure that the device has booted in updater mode
3. Read updater version (optional).
4. Read updater buffer size to know the maximum data block that can be written in a single aci_updater_program_data_block command.
5. For BlueNRG only, dump the IFR sector using aci_updater_read_data_block command
6. Erase IFR sector 32
7. For BlueNRG only, write the dumped IFR sector except for the last 192 bytes, where IFR configuration is stored.
8. Write IFR configuration in the last 192 bytes of IFR sector by using aci_updater_program_data_block command.
9. Verify that the sector has been correctly programmed. aci_updater_read_data_block command can be used to read sector content.
10. Reset the device to reboot the system. It is also possible to switch off the power.

9 Revision history

Table 26. Document revision history

Date	Revision	Changes
29-May-2014	1	Initial release.
18-Jan-2016	2	Added reference to BlueNRG-MS device.
19-Dec-2016	3	Updated *Section 7: Firmware update algorithm*, added new *Section 8: IFR update algorithm* and BlueFlag address location.