MediaTek LinkIt™ Development Platform for RTOS Firmware Update Developer's Guide

Version:	1.0
Release date:	07 July 2018

Document Revision History

Revision	Date	Description
1.0	07 July 2018	Initial release.

Table of Contents

Lists of Tables and Figures

1. Overview

1.1. FOTA Introduction

MediaTek MT2625 platform enables firmware over the air (FOTA) update, a widely adopted cost and time efficient solution to update the firmware on connected devices. The purpose of the developer's guide is to provide complete description on how to deploy FOTA on the NOR flash.

This document guides you through:

- FOTA architecture overview
- FOTA settings
- FOTA package tools
- FOTA download agent
- FOTA upgrade agent

A complete list of FOTA features for MediaTek MT2625 development platform for RTOS is provided below.

- Full binary update
- FOTA packaging tool
- Trigger FOTA by LWM2M
- Security FOTA

By default, the SDK provides full binary update mechanism. The security FOTA feature is not enabled. The SDK also enables third party solution integration to provide incremental update mechanism.

1.2. FOTA Architecture Overview

FOTA feature mainly can be divided into three parts: FOTA upgrade agent, FOTA packaging tool and FOTA download agent. FOTA architecture overview is shown in Figure 1.

- The **FOTA Upgrade Agent** is part of the **Bootloader** and it uses flash APIs to process the update on the target device. While the device is connected to the NB-IoT network, FOTA AT commands can be used to download the FOTA package files from HTTP server.
- The **FOTA packaging tool** (`FOTARomPacker.exe` located under root folder of the package) is used on the PC to apply the Lempel–Ziv–Markov chain algorithm (LZMA) compression and signature in the new firmware file and to generate a FOTA update package. Due to MT2625's flash space limit, MT2625 only supports compressed type for MT2625 firmware binary.
- The **FOTA Download Agent** is part of FOTA task. It is responsible for downloading the FOTA package from remote server. Currently, the FOTA package is downloaded by HTTP protocol. While the device is connected to the NB-IoT network, FOTA AT commands can be used to download the FOTA package files from HTTP server. Or, the download process can be triggered by LWM2M server.

The numbered items on Figure 1 are explained below:

1) **FOTA packaging tool** — generates an update package.

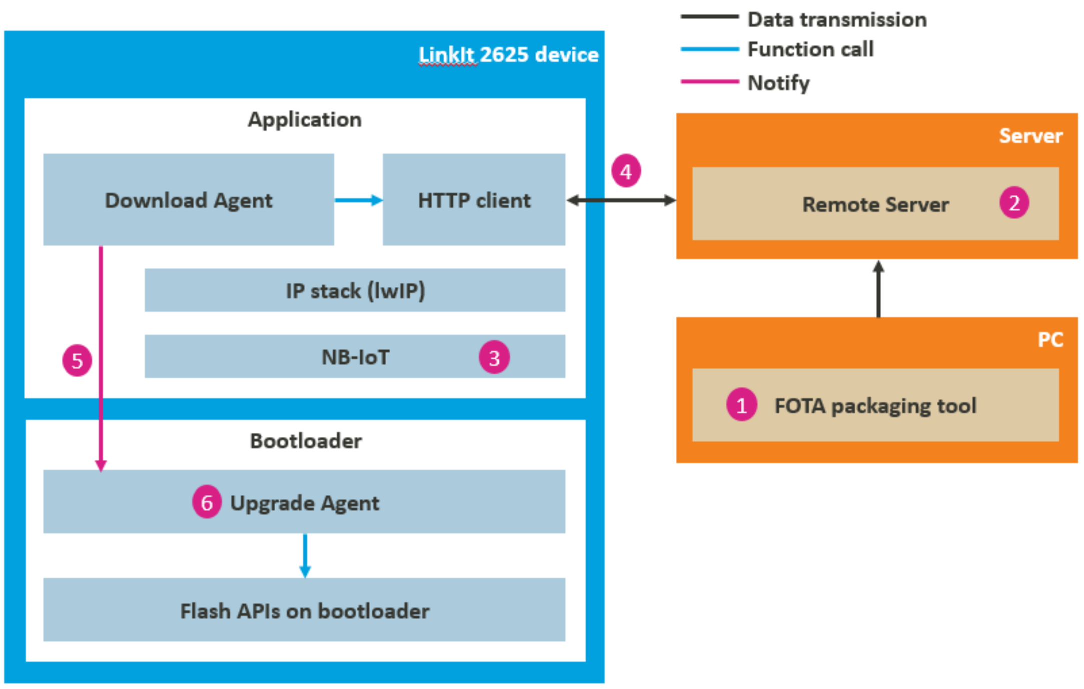
2) **Remote server** — puts the update package to the remote server.

3) **NB-IoT** — the device connects to the network.

4) **Download Agent** — Download the FOTA package from remote server.

5) Sets the FOTA triggered flag and reboots the device.

6) **Upgrade Agent** — checks the flag and proceeds with binary upgrade.



1. Generate an "updated package" with FOTA packaging tool.
2. Put the "updated package" to the server. (HTTP server)
3. MT2625 based device connects to the NB-IoT network.
4. Download the "updated package" using HTTP.
5. After the download is complete, it set the triggered flag and reboots the device.
6. The update agent checks the flag and starts the update process.

Figure 1. FOTA architecture overview

2. FOTA Settings

To enable FOTA related functions, for specific requirements, specific FOTA settings must be set up, such as compile options, FOTA reserved memory layout, etc.

2.1. Compile options for the full binary FOTA update

The compile options for the FOTA are configured in the main project and bootloader project makefiles (`feature.mk`). One of the makefiles is in the main application folder, the other is in the bootloader project folder that builds the bootloader binary. Only when FOTA options in both of them are enabled, the FOTA feature is enabled.

Configure the following option in `PROJECT_DIRECTORY/GCC/feature.mk` of the main application.

```
MTK_FOTA_ENABLE = y
```

The same option also needs to be configured for the bootloader's makefile under `project/mt2625_evb/apps/bootloader/GCC/feature.mk` in bootloader project folder, as shown below:

```
MTK_FOTA_ENABLE = y
```

If security FOTA is enabled on the device, other compile options must be enabled in the `PROJECT_DIRECTORY/GCC/feature.mk` file, as shown in Table 1.

Table 1 . Security FOTA related compile options

Option	Description	Build the ARM Cortex-M4	Build the bootloader
MTK_SECURE_BOOT_ENABLE	Enable security boot feature	=y	=y
MTK_BOOTLOADER_USE_MBEDTLS	Enable mbedtls library	=y	=y

If MTK GNSS chip is connected with MT2625, to enable GNSS FOTA feature, below compile option must be enabled under `PROJECT_DIRECTORY/GCC/feature.mk.`

```
MTK_GNSS_FOTA_ENABLE = y
```

2.2. FOTA buffer partition size for full binary update

The FOTA update package data is saved in the FOTA reserved partition. To customize the flash layout, consider saving the storage size for the FOTA reserved partition. Usually FOTA packaging tool will compress the binary file to reduce the reserved quota size by default, so the reserved storage is 65% of the total size of the target partitions ready to update.

2.3. Flash layout settings for FOTA feature

Default flash layout with FOTA enabled on MT2625 is shown in Figure 2.

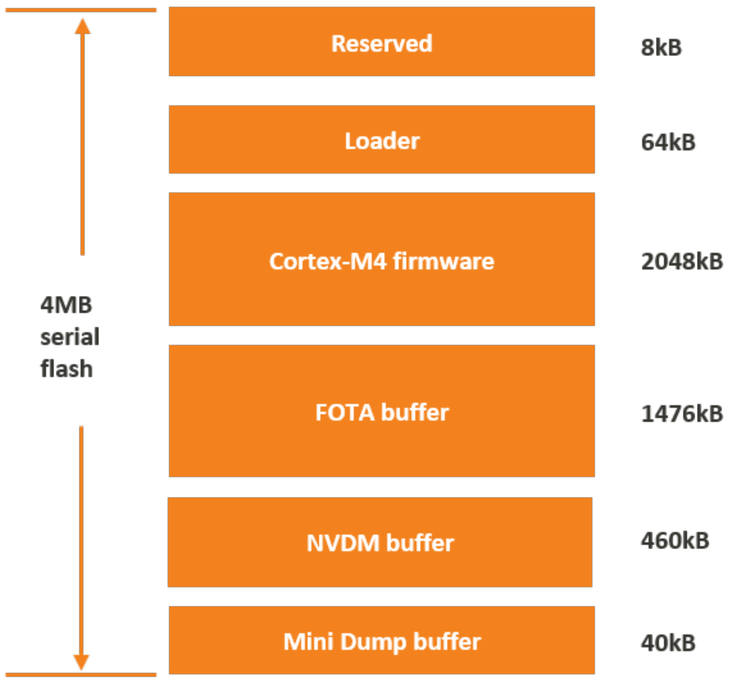


Figure 2. MT2625 flash layout default settings

Description on how to change the FOTA reserved partition size can be referred to MT2625 memory layout development guide.

3. FOTA packaging tools

MediaTek MT2625 platform provides a FOTA packaging tool that runs on Microsoft Windows and Linux OS to compress data, generate a checksum and prefix header for the new binary FOTA package file. During the upgrade, the upgrade agent parses the header, validates the package file using the checksum and identifies where the data should be written.

3.1. Microsoft Windows version of the FOTA packaging tool

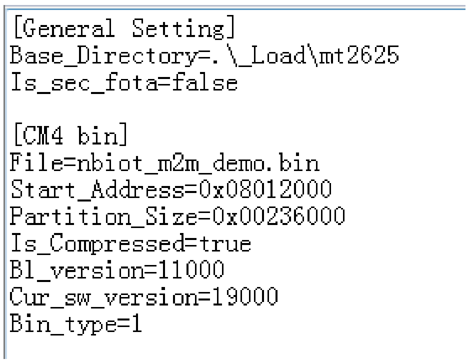
Windows version of the FOTA packaging tool consists of the following items shown in Table 2

Table 2. The FOTA packaging tool's folder content on Windows PC

Folders and Files	Description
`_ini`	Contains configuration and initialization files
`_Load`	A folder for a new image
`_Output`	A folder with output package files.
`FOTARomPacker.exe`	The main executable file to parse configuration files, header prefix, compress the data and generate a checksum.
`gen_image.bat`	Batch file with a complete configuration.

To use the FOTA packaging tool:

1) Copy the new binary to the _Load folder.
2) Configure the `FOTARomPacker.ini` file located under `_ini` folder.
   - Set the load path to the line below `[General Setting]` (see Figure 3).
   - If the security boot feature is enabled, set security filed (Is_sec_fota) as true.
   - Set the binary file's name (`File`) and the start address (`Start_Address`) of this binary in the flash.
   - Set the size of the partition where the binary will be written.
   - Configure the setting to compress the binary if needed. The LZMA compression is applied, if the value of `Is_Compressed` is `true`.
   - Set the bootloader's version by field: Bl_version.
   - Set the MT2625 firmware's version by field: Cur_sw_version.
   - Set the binary's type by field: Bin_type.
3) Execute the `gen_image.bat` batch file to generate a FOTA package file under `_Output` folder.



```
[General Setting]
Base_Directory=.\_Load\mt2625
Is_sec_fota=false

[CM4 bin]
File=nbiot_m2m_demo.bin
Start_Address=0x08012000
Partition_Size=0x00236000
Is_Compressed=true
Bl_version=11000
Cur_sw_version=19000
Bin_type=1
```

Figure 3. FOTARomPacker.ini FOTA configuration file content

3.2. Using the Linux version of the FOTA packaging tool

The Linux version of the FOTA packaging tool consists of the items shown in Table 3.

Table 3. The FOTA packaging tool's folder content on Linux PC

Folders and Files	Description
`config_template`	Contains configuration files in JSON format for FOTA configurations.
`example`	An example with configuration, binary, executable and other files to generate the FOTA update file. The files should be replaced for other HDK configuration.
`src`	The source code of the tool.
`build.linux.sh`	The source should be compiled by `cmake` compiler on Linux OS to generate an executable named "`BinBuilder`" for the FOTA packaging tool. (If `cmake` compiler is installed on Windows OS, use `build.windows.bat`.)
`CMakeLists.txt`	Makefile for `cmake` compiler.
`README.md`	Describes how to use the tool.

To use the FOTA packaging tool, configure the JSON configuration file with the following attributes:

Table 4. Configuration attributes

Attribute		Type	Description
`output`		string	Defines the output configuration file path.
`Is_sec_fota`		bool	Defines whether security FOTA is enabled.
`bins` (An array of dictionary with 1 to 4 items)	`file_path`	string	Path to the binary file.
	`start_addr`	string	Offset address of the binary file in the flash, such as "`0x7B00`".
	`partition_size`	string	Size of the binary file on the flash, such as "`0x8000`".
	`is_compressed`	bool	"`true`", if the binary file was compressed before packing.
	`bl_version`	int	The current bootloader's version.
	`cur_sw_version`	int	The firmware's software version.

Attribute		Type	Description
	Bin_type	int	The current binary's type. Main binary type is 1. Security header binary type is 2. GNSS binary type is 3.

Each item in "bins" represents a binary file with attributes listed in Table 4. For example, there are two binary files "nbiot_m2m_demo.bin" and "mt2625_evb_cm4_hdr.bin" with file path, start address, partition size and compression status, as shown below.

Note, that "start_addr" and "partition_size" are strings instead of numbers to accept C-style hexadecimal literals, such as "0xFFEF". An example configuration file looks like this:

```
{
  "output" : "image_sec.bin",
  "is_sec_fota": true,
    "bins": [
    {
      "file_path": "nbiot_m2m_demo.bin",
      "start_addr": "0x08012000",
      "partition_size": "0x00236000",
      "is_compressed": true,
      "bl_version": 11000,
      "cur_sw_version": 12000,
      "bin_type": 1
    },
    {
      "file_path": "mt2625_evb_cm4_hdr.bin",
      "start_addr": "0x08001000",
      "partition_size": "0x00001000",
      "is_compressed": false,
      "bl_version": 11000,
      "cur_sw_version: 12000,
      "bin_type": 2
    }
  ]
}
```

By default the program loads "config.json" in the same directory of the executable file ("BinBuilder"). Alternatively, JSON configuration file path can be assigned through command parameter, as shown below:

```
./BinBuilder my_config.json
```

4. FOTA Download Agent

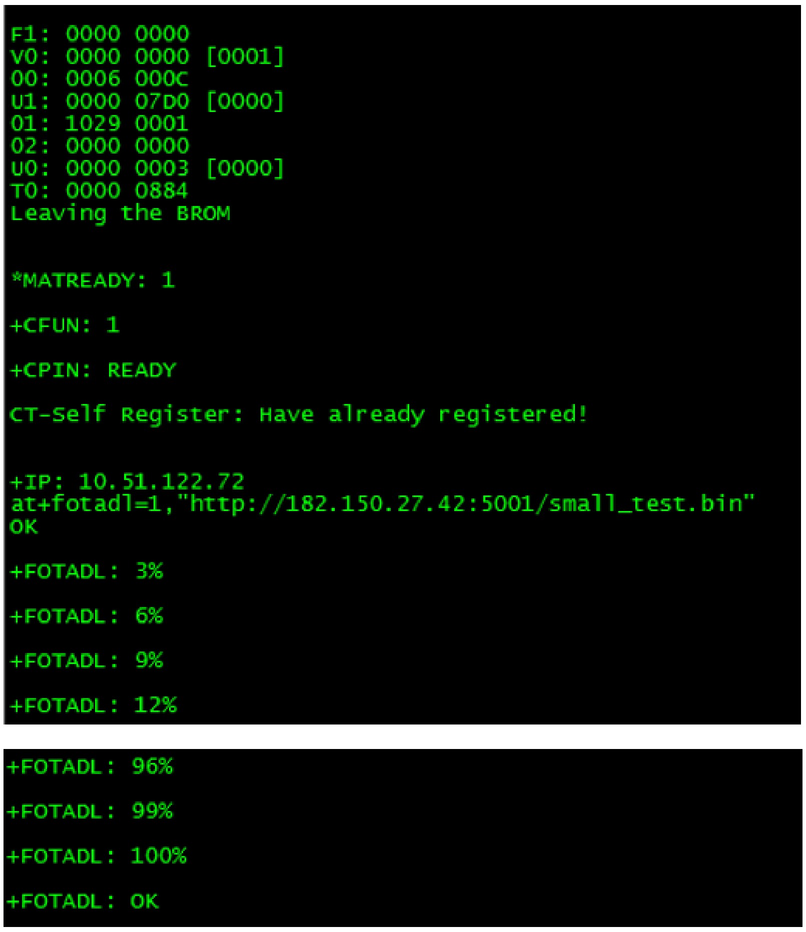
When the FOTA package is generated, the FOTA package will be putted into remote server. Then, FOTA download agent can download the FOTA package by NB-IoT network with different protocol. In MT2625 platform, FOTA package is downloaded by HTTP protocol. FOTA download agent is a part of FOTA task in middleware. Applications can call the FOTA's interface functions to trigger FOTA download agent to download FOTA package. Currently, in MT2625 development platform, download can be triggered by AT command and LWM2M firmware object.

4.1. Trigger FOTA download by AT command

The user can input AT command to trigger the FOTA package download process. All the FOTA packages are downloaded by command "AT+FOTADL". For more detailed information about this command, please refer to MT2625 none-modem AT command specification.

- Download MT2625 firmware FOTA package
  AT+FOTADL=1,"<MT2625 FOTA package URL>"
- Download GNSS firmware FOTA package
  AT+FOTADL=2,"< GNSS FOTA package URL >"

When the download processor is triggered, the package's download percent is shown in the AT command's port. The download example is shown in Figure 4.



```
F1: 0000 0000
V0: 0000 0000 [0001]
00: 0006 000C
U1: 0000 07D0 [0000]
01: 1029 0001
02: 0000 0000
U0: 0000 0003 [0000]
T0: 0000 0884
Leaving the BROM

*MATREADY: 1

+CFUN: 1

+CPIN: READY

CT-Self Register: Have already registered!

+IP: 10.51.122.72
at+fotadl=1,"http://182.150.27.42:5001/small_test.bin"
OK

+FOTADL: 3%

+FOTADL: 6%

+FOTADL: 9%

+FOTADL: 12%
```

```
+FOTADL: 96%

+FOTADL: 99%

+FOTADL: 100%

+FOTADL: OK
```

Figure 4. FOTA Package Download

4.2. Trigger FOTA download by LWM2M

Precondition

The device has been connected to LWM2M server

Trigger FOTA download steps

- In LWM2M server, finding the corresponding device.
- Fill the FOTA package's HTTP URL into LWM2M package URI field.
- When the device receives FOTA package's URI in LWM2M firmware object, firmware object will trigger FOTA task to download the FOTA package. The FOTA package's download status will be synchronized to LWM2M server in real time.

5. FOTA Upgrade Agent

This section provides details on how to download the FOTA package and update the firmware in bootloader on MT2625 development platform.

5.1. MT2625 Firmware Upgrade Agent

The purpose of the Upgrade Agent is to read the FOTA binary that contains the MT2625 firmware binary and then write the binary to the MT2625 firmware partition. The workflow of Upgrade Agent is shown in Figure 5.

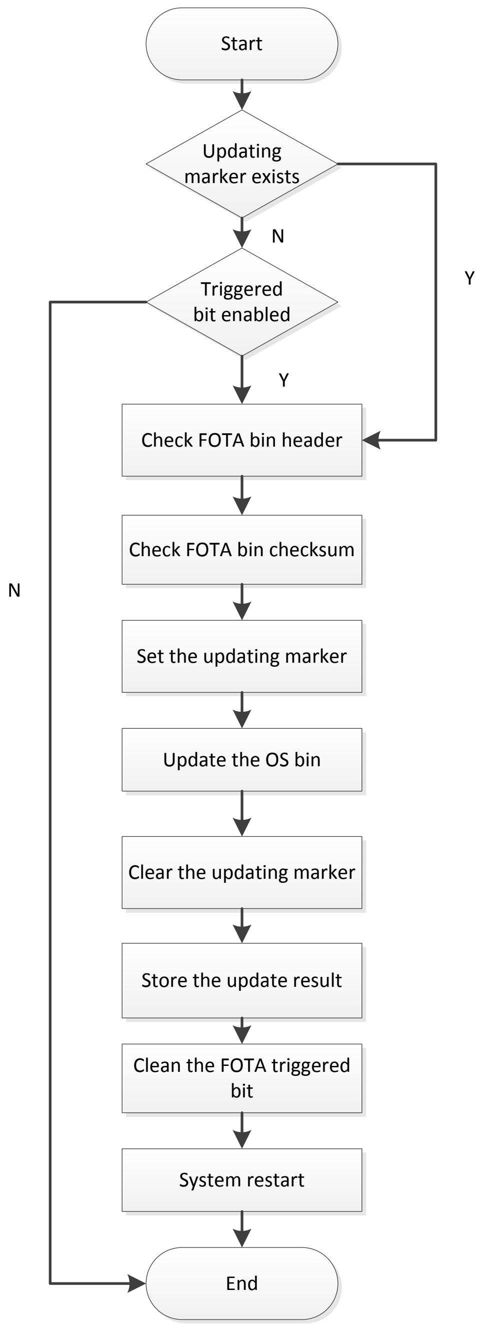


Figure 5. Upgrade Agent Flow

The FOTA binary includes a header, the MT2625 firmware binary and a checksum. The **Upgrade Agent** reads the header to identify the start address and size of the firmware partition. The header also includes the checksum position and a magic number. The **Upgrade Agent** uses the magic number to check if the FOTA binary is available and uses the checksum to confirm the FOTA binary contains no errors. The FOTA binary header is provided in the `IOT_FOTA_HEADER` structure in the header file located at `PROJECT_DIRECTORY /apps/bootloader/ /inc/bl_fota.h`, as shown below.

```
typedef struct {
    uint32_t m_bin_offset;
    uint32_t m_bin_start_addr;
    uint32_t m_bin_length;
    uint32_t m_partition_length;
    uint32_t m_sig_offset;
    uint32_t m_sig_length;
    uint32_t m_is_compressed;
    uint32_t m_version_info[16];
    uint32_t m_bin_type;
    uint8_t m_bin_reserved[4];
} bl_fota_bin_info_t;


typedef struct {
    uint32_t m_magic_ver;
    uint32_t m_bin_num;
    bl_fota_bin_info_t m_bin_info[BL_FOTA_BIN_NUMBER_MAX];
  } bl_fota_header_t;
```

The **Upgrade Agent** decides whether to start the FOTA update flow based on the value of the FOTA triggered bit in the retention register, once the bootloader starts.

There is a power failure protection for the MT2625 firmware upgrade. Before the firmware binary update, the upgrade agent writes a marker to the flash, as shown in Figure 6. After the MT2625 firmware binary upgrade is complete, the marker is cleared from the flash. If power failure occurred during the MT2625 firmware upgrade, the **Upgrade Agent** checks marker to restart the process.

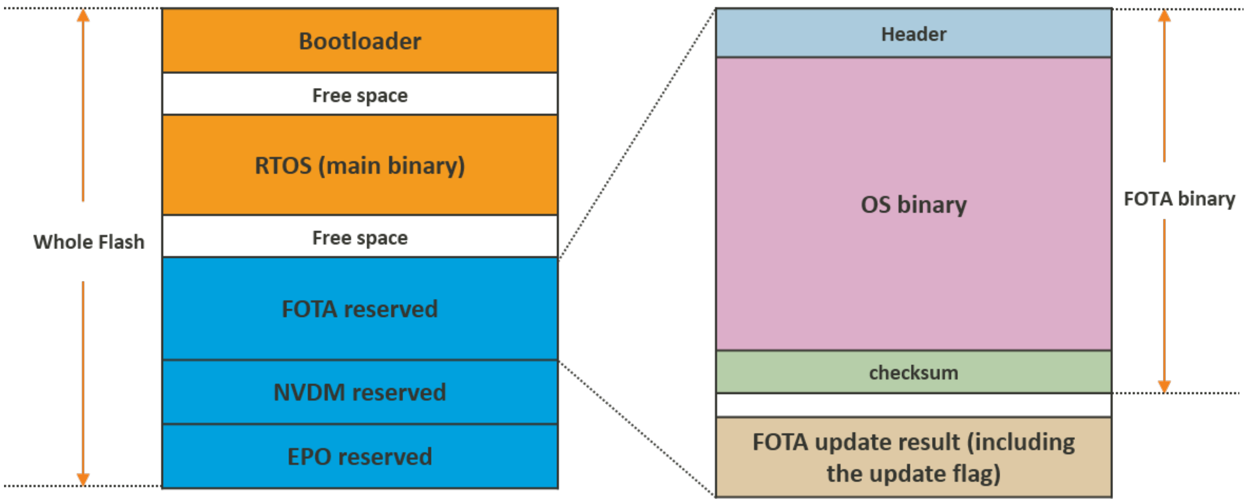


Figure 6. FOTA binary reserved memory

The FOTA binary is stored in the FOTA buffer partition. The FOTA buffer partition contains the binary and the final update result. The FOTA binary is written using the Download Manager and the update result is written by the Update Agent in bootloader. The Download Manager then sends the update result to the smart device application. Then the application displays the corresponding string to notify the end user.

The FOTA binary is placed at the tail of the FOTA buffer partition, as it only occupies a block size and is smaller compared to the FOTA buffer partition size. FOTA update result is stored in the tail of the FOTA Buffer partition. Once the update is finished, the Download Manager reads the result and then erases the tail of FOTA buffer partition.

5.2. GNSS Firmware Upgrade Agent

GNSS Firmware upgrade agent can be divided into two parts. For MT2625 platform, we only focus on the agent on MT2625 side. The GNSS upgrade agent on MT2625 locates in FOTA task. The code can be found shown as below path: middleware\MTK\fota\src\internal\gnss_fota. GNSS FOTA upgrade flow is shown in Figure 7.

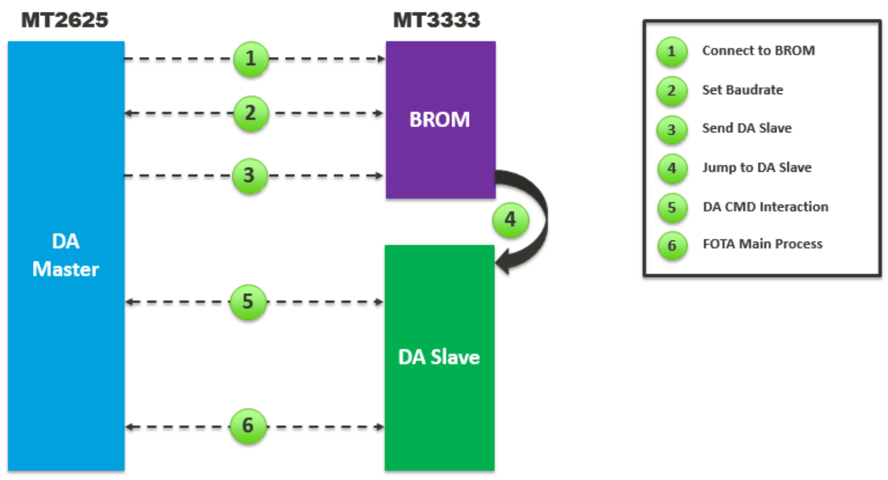


Figure 7 . GNSS FOTA Upgrade Agent Flow

Note: The SDK FOTA update on MT2625 mainly focuses on GNSS and ARM Cortex-M4 firmware, bootloader partition cannot be updated.

6. Appendix A: Acronyms and Abbreviations

The acronyms and abbreviations used in this developer's guide are listed in Table 5.

Table 5. Acronyms and abbreviations

Abbreviation/Term	Expansion/Definition
FOTA	Firmware over the air, a way to update firmware wireless
Full Binary	A type of FOTA to update the whole firmware.
NOR flash	NOR flash is a type of non-volatile storage that does not require power to retain data.
NB-IoT	Narrowband Internet of Things