MT793X IoT SDK for Secure Boot User Guide

Version: 0.5
Release date: 2022-10-06

Version History

Version	Date	Description
0.1	2021-04-06	Initial draft
0.2	2021-04-06	Add Terminology Correct Figure
0.3	2021-04-07	Update Figure 2-5 Add Q&A chapter Revise description in Section 2.3
0.4	2022-07-18	Update for EDCSA P384/P521
0.5	2022-10-06	Update section 2.3 for anti-rollback description

Table of Contents

List of Figures

List of Tables

Terminology

Items	Description
AES-OTF	AES On The Fly
BL	Boot Loader
BROM	Boot ROM. Root of trust in MediaTek chips.
DA	Download Agent. A software downloaded by BROM for writing flash
DAA	Download Agent Authentication
DAA_PUBK	DAA public key
DAPC	Device Access Permission Control
EFUSE	One Time Programmable Memory
KDF	Key Derivate Function
SBC	Secure Boot Chain
SBC_PUBK	SBC public key (Root Public Key)
SBC_PRIK	SBC private key
SBC_PUBK_HASH	The hash of SBC public key
SCTRL_CERT	Secure Control Certificate File
SEJ	Secure Engine with JTAG
SMPU	System Memory Protection Unit
TLV	Type-Length-Value format
TOOL_AUTH	Tool Authentication File. It is used in secure download

1 Overview

BootROM is the first stage of bootup; it takes control when the CPU exits from reset. The main tasks of BootROM are boot and handle firmware download. This document introduces these functionalities.

The types of security supported in BROM are categorized as below:

- Booting security
    - SBC (Secure Boot Chain): Verify the signature of the bootloader.
    - Anti-rollback: Prevent the bootloader from rolling back to old versions with issues.
    - AES-OTF: AES on the fly to prevent cloning of the bootloader.
- Debug security
    - SCTRL_CERT: A certificate used for controlling DAA, and JTAG debug port policy.
- Tool security
    - TOOL_AUTH: Tool authentication file for successive tool operation.
    - DAA (Download Agent Authentication): Authenticate the download agent.

These features are illustrated in Figure 1-1.

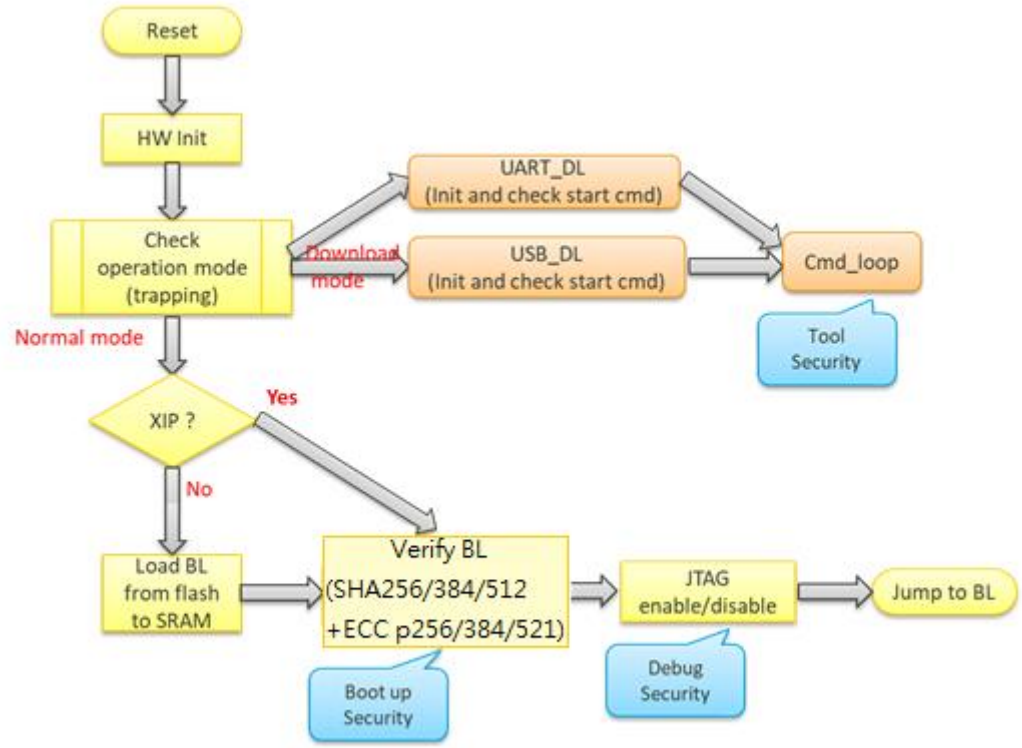

Figure 1-1 BootROM secure boot feature

2 Boot up Security

2.1 SBC Boot Up

The main task of BootROM is booting up. In Figure 2-1 BootROM boot flow, when the CPU exits from reset, BootROM takes over the responsibility for setting up C execution environment and basic hardware configuration such as UART, timer, watchdog.

BootROM then checks the trapping to get the mode you choose to enter. If the trapping value is normal boot, BootROM initializes the serial flash and parses the bootloader's information from header. Thus, BootROM can know where to execute the bootloader since both execute in place and boot from RAM are supported.

If the load address in the header of the bootloader is flash based address, it means execute in place (XIP) and BootROM verifies the bootloader on serial flash directly.

If the load address in the header of the bootloader is SRAM based address, it means executing the bootloader on SRAM, and BootROM loads the bootloader from serial flash to SRAM and verifies the copy of bootloader on SRAM.

Once the bootloader passes the signature verification, BootROM trusts the bootloader and then parses the private TLV to enable or disable JTAG

Now that everything is ready, BootROM jumps to the bootloader and the mission is completed.

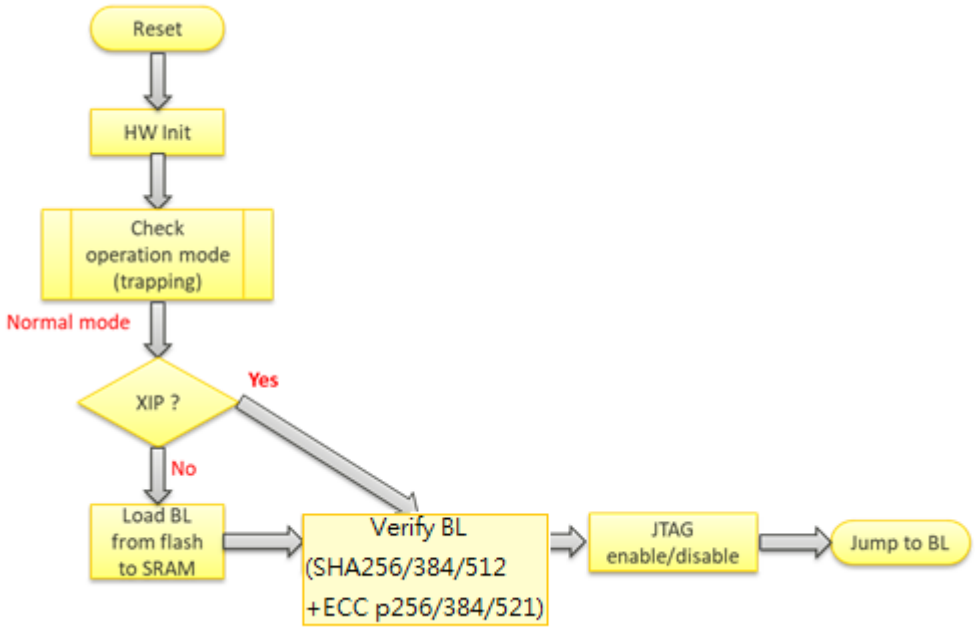

Figure 2-1 BootROM boot flow

2.2 Verification Flow

BootROM checks the integrity and authenticity of the bootloader. To enable secure boot checking, you need to blow eFuse as shown In Table 2-1.

Table 2-1 eFuse to enable secure boot

Item	Description
EFUSE_SBC_EN	Enable secure boot chain
EFUSE_SBC_ALGO	SBC algorithm option (ECDSA-P256, ECDSA-P384 or ECDSA-P521)
EFUSE_SBC_PUBK_HASH0	HASH of public key 0 for secure boot
EFUSE_SBC_PUBK_HASH1	HASH of public key 1 for secure boot
EFUSE_SBC_PUBK_HASH0_DIS	Disable HASH of public key 0
EFUSE_SBC_PUBK_HASH1_DIS	Disable HASH of public key 1

Before introducing how BootROM verifies the bootloader, this section talks about the composition of bootloader image as shown in Figure 2-2.

In the compiler time of the bootloader, the image packing/signing tool adds header and appends some properties called “Protected TLV” in the tail that you can specify, and then the image tool calculates the contents including “header”, “bootloader payload”, and “Protected TLVs” with SHA-256/SHA-384/SHA-512, and signs the HASH value with ECDSA p-256/p384/p521, and then pads the public key and signature to the tail in TLV format. This part is called “public TLV”.

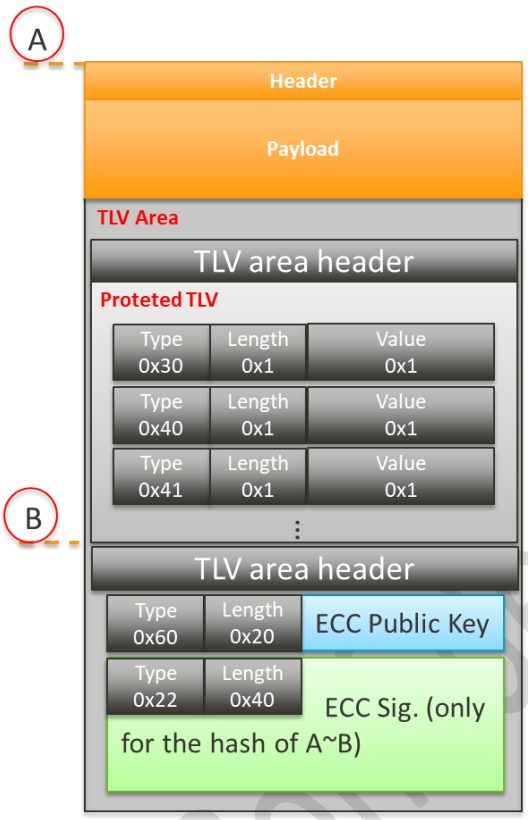

Figure 2-2 Composition of bootloader image

Hashing algorithm	SHA-256
Digital signature algorithm	ECDSA p-256
Hashing algorithm	SHA-384
Digital signature algorithm	ECDSA p-384
Hashing algorithm	SHA-521
Digital signature algorithm	ECDSA p-512

Next, BootROM verifies the bootloader, as shown in Figure 2-3.

First, BootROM reads the header of bootloader image to know the header size and bootloader payload size; thus, BootROM can calculate the location of protected TLV and public TLV, and calculate HASH value in advance.

Second, BootROM gets the public key from public TLV and calculates the HASH of key, and compares it with the key HASH stored in eFuse (that means you must blow the HASH value of the public key into eFuse)

Third, now you have HASH value and valid public key, feed them to ECC hardware engine for verification, and the hardware returns the result of signature verification.

After the three steps above, BootROM recognizes this bootloader is valid, and parses the protected TLV in advance.

If an error occurs in any step, the boot-up procedure fails.

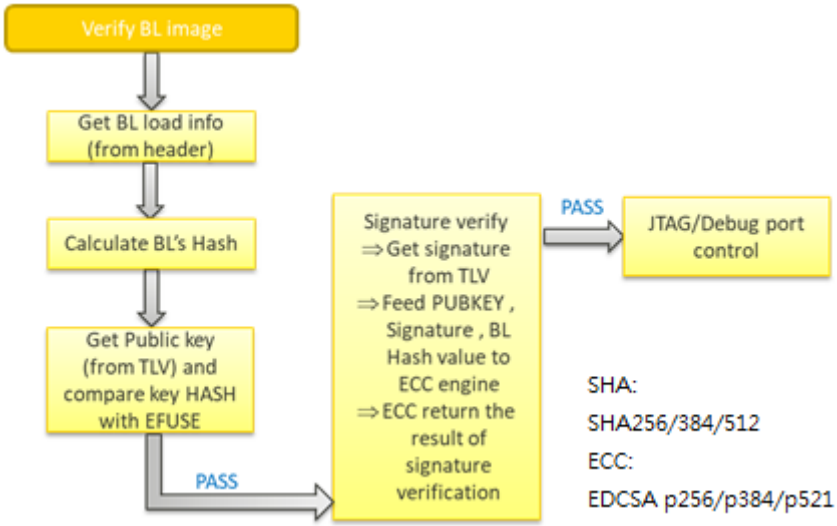

Figure 2-3 BootROM verify bootloader

2.3 Anti-rollback

MT793X BootROM supports anti-rollback. You can blow eFuse to update the valid bootloader version. Set the EFUSE_PL_AR_EN bit to enable anti-rollback and set the 64-bit eFuse bits to have 64 versions bootloader anti-rollback checking as shown in Table 2-2.

Table 2-2 eFuse for anti-rollback

Item	Description
EFUSE_PL_AR_EN	Enable bootloader anti-rollback
EFUSE_PL_VER0/1	Bootloader version bits

After the bootloader is verified, BootROM parses the bootloader's anti-rollback version in protected TLV and compares the value with eFuse. If the bootloader's anti-rollback version **is bigger than or equal to** the version written in eFuse, the checking completes and the boot-up procedure continues; otherwise, the boot-up procedure fails. Figure 2-4 shows the TLV format used for anti-rollback version, and the comparison logic. Figure 2-5 illustrates the BootROM anti-rollback flow.

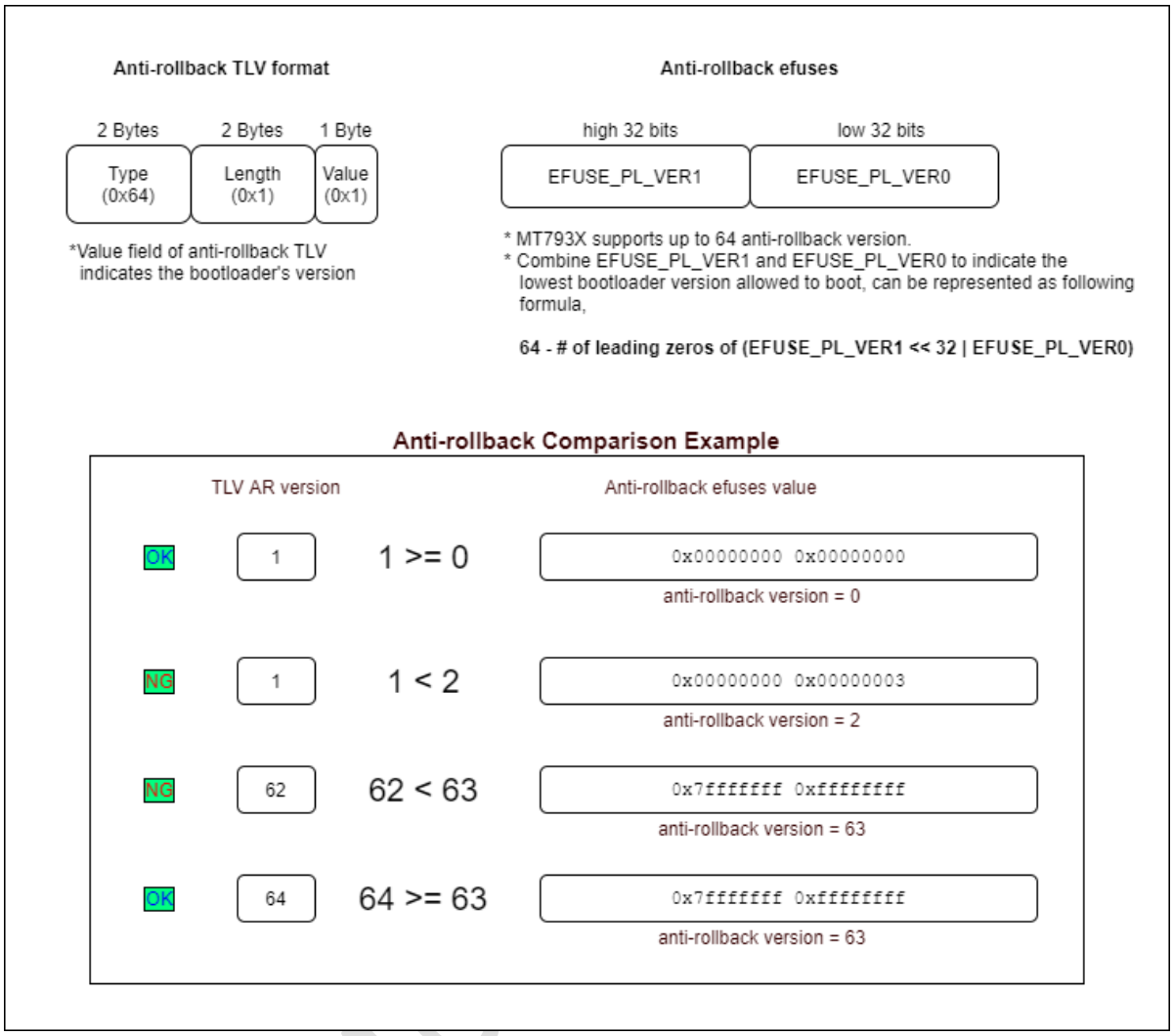

Figure 2-4 Anti-rollback TLV Format, Efuses and Comparison Logic

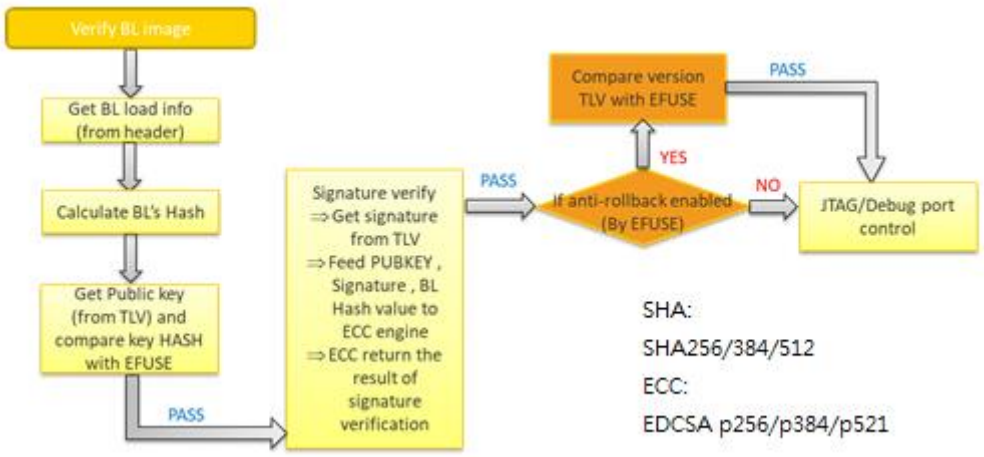

Figure 2-5 Anti-rollback check flow

2.4 AES-OTF

BootROM supports AES-OTF (AES On-The-Fly). The objective of this feature is to encrypt serial flash data to prevent plaintext leak, and automatically decrypt the ciphertext to plaintext when accessing data from serial flash (read operation).

You can blow eFuse to enable AES-OTF feature in BootROM, blow eFuse to select key source, and blow eFuse for NONCE, as shown in Figure 2-6

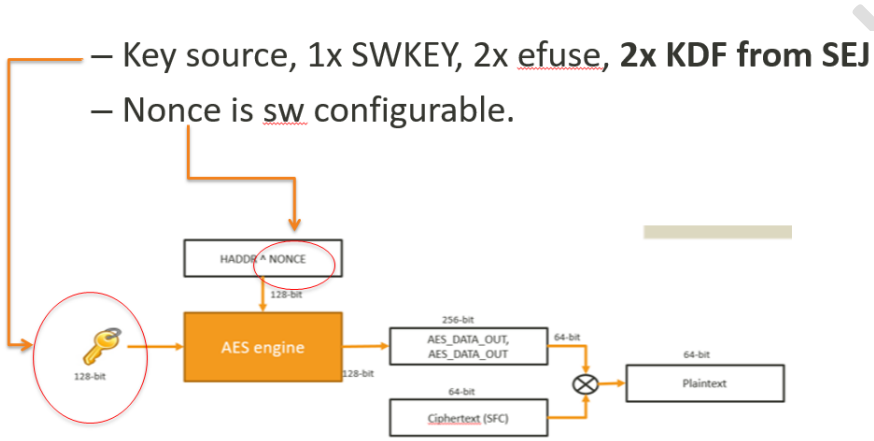

Figure 2-6 Input/Output of AES-OTF

Table 2-3 provides a list of related eFuses, and Figure 2-8 shows the logic to apply key source and nonce.

Table 2-3 AES-OTF eFuse

Item	Description
EFUSE_AES_OTF_EN	Enable BROM's AES-OTF
EFUSE_AES_OTF_KEY_SOURCE	Key source selection 0: Use EFUSE_AES_OTF key0 1: Use EFUSE_AES_OTF key1 2: Use software constant key 3: Use SEJ_KDF key 0 4: Use SEJ_KDF key 1
EFUSE_AES_OTF_PERDEV_EN **EFUSE_AES_OTF_OTP**	Use per-device OTP as NONCE
EFUSE_AES_OTF_KEY0	EFUSE_AES_OTF key 0
EFUSE_AES_OTF_KEY1	EFUSE_AES_OTF key 1
EFUSE_HUID	Hardware unique ID

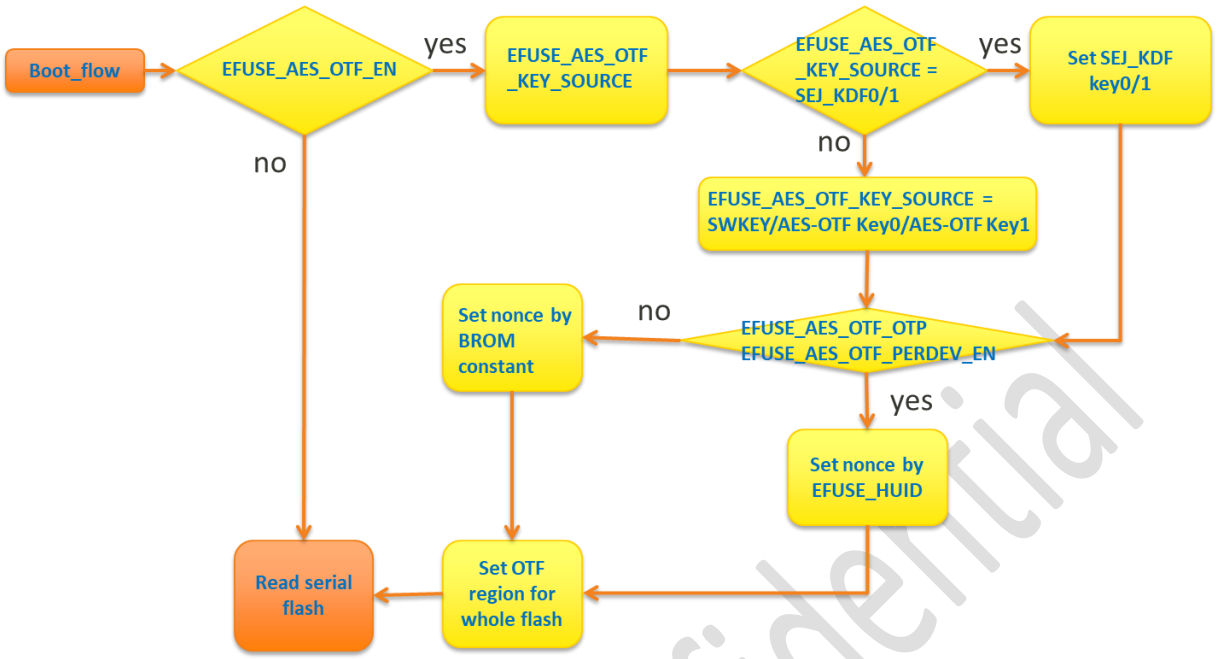

Figure 2-7 AES-OTF key and nonce selection

Figure 2-8 shows the AES-OTF configuration in secure boot flow.

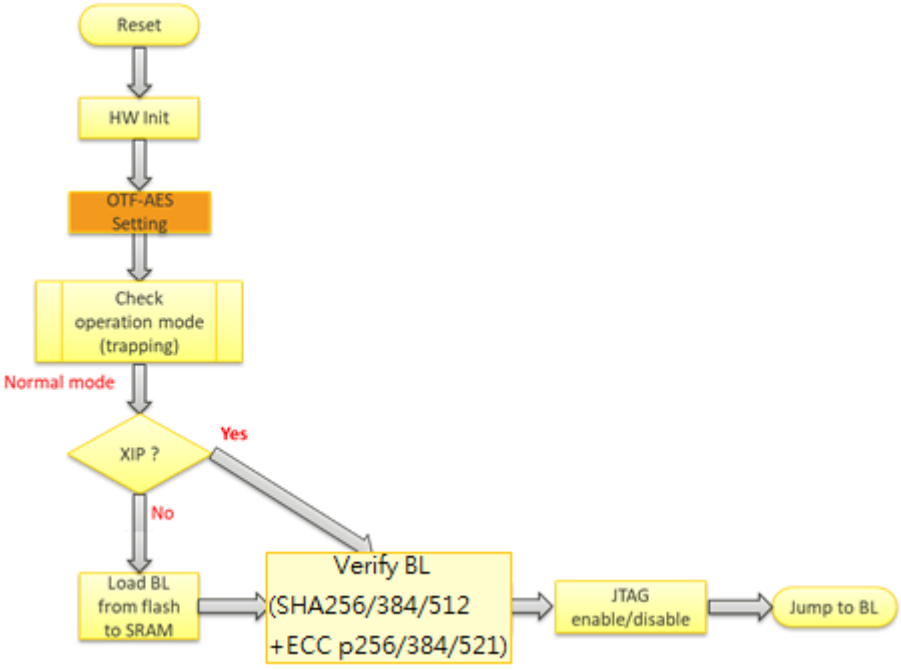

Figure 2-8 AES-OTF configuration in boot flow

2.5 Bootloader Image File Layout

As shown in Figure 2-9, you can append TLVs like SMPU configure, DAPC configure and Audio DAPC configure; they are address-value pairs that are dedicated to SMPU and DAPC setting in BootROM stage, according to product requirement.

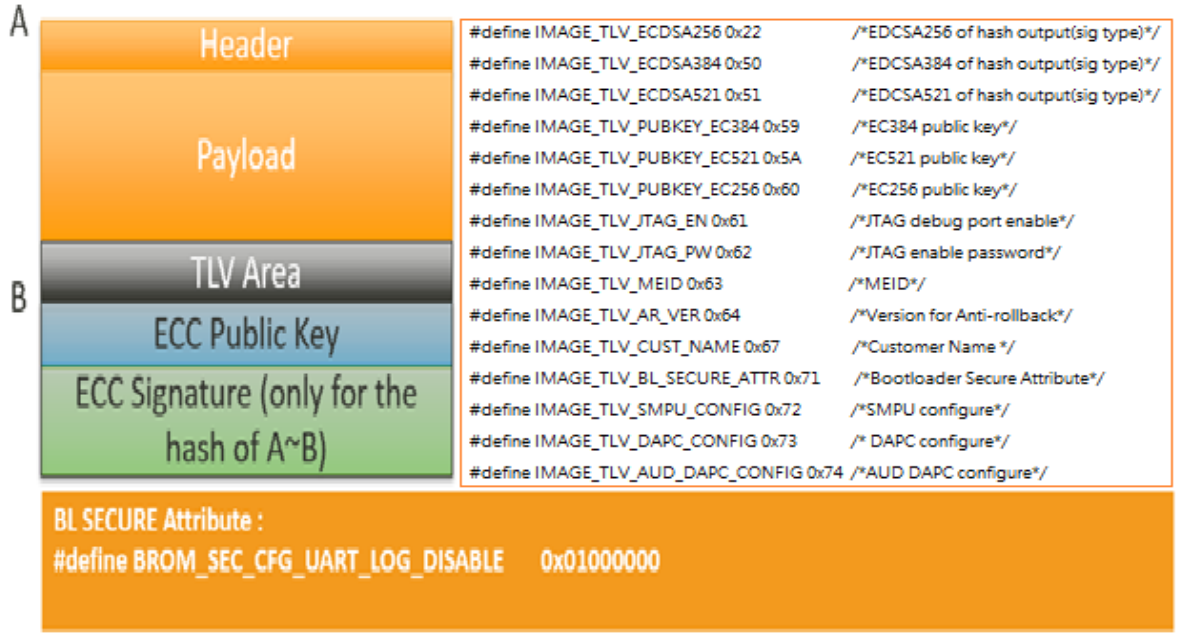

Figure 2-9 Bootloader image file layout

3 Debug Security

The MT793X supports debug port that communicates with external PC tool. Debug port can be configured through per-device certification; BootROM can use ME_ID and SCTRL_CERT to identify the user.

3.1 MEID

ME_ID is a bind-to-chip identity generated inside BootROM, stored in TCM and immutable by the device. Figure 3-1 shows when to generate ME_ID in BootROM boot up flow.

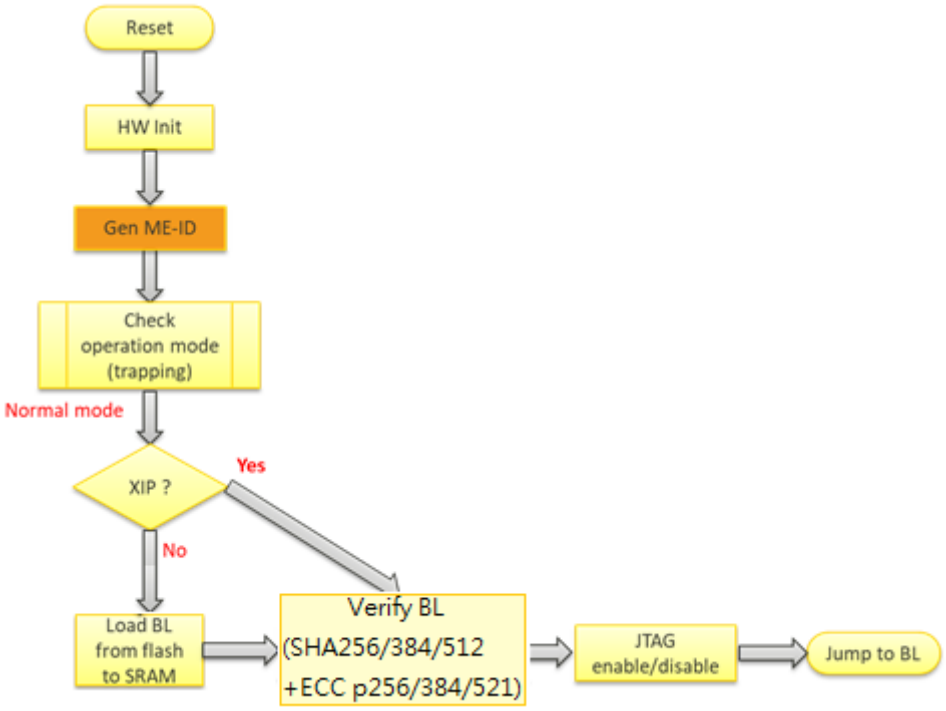

Figure 3-1 Gen MEID in boot up

3.2 SCTRL_CERT

SCTRL_CERT (secure control certification) is a certification supporting debug option, which can be used to

1. Disable anti-rollback for bootloader and tool authentication file

2. Enable JTAG for debug purpose

This certification file can be downloaded in BootROM download mode, and BootROM verifies the certification file the way it verifies the bootloader. If the certification file is valid, this certification file is kept in TCM and disappears when the device is powered off.

For JTAG configuration, it works only after the device is reset.

And for anti-rollback disabling, it can take effect on tool authentication directly, and take effect for bootloader after the device is reset.

Figure 3-2 and Figure 3-3 show the SCTRL_CERT work flow.

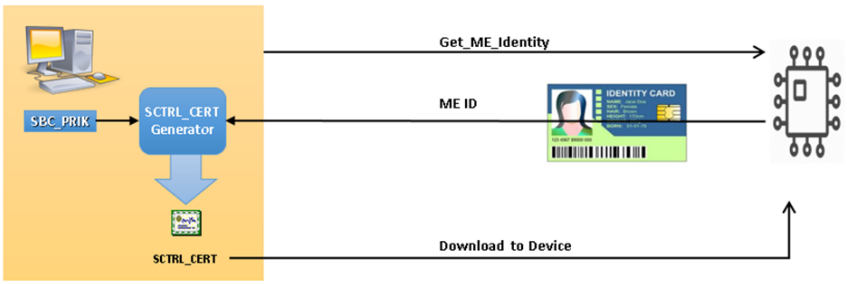

Figure 3-2 How SCTRL_CERT works

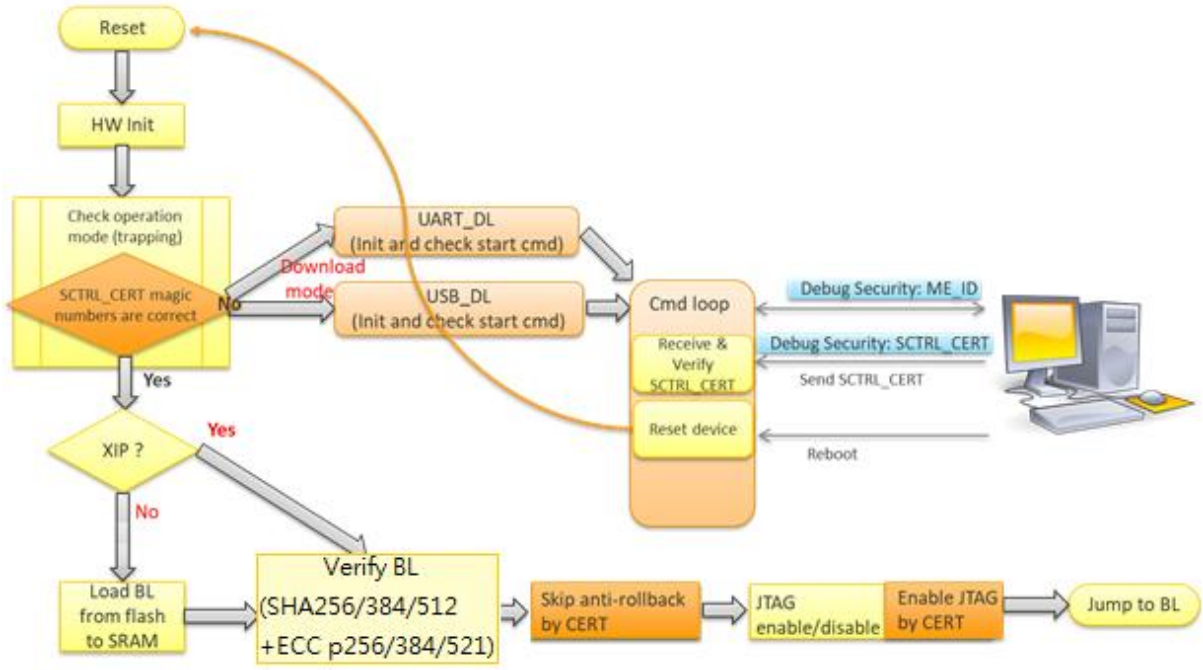

Figure 3-3 SCTRL_CERT work flow

In MP IC, you can choose to blow EFUSE_SW_JTAG_CON to disable debug port by default or choose to enable debug port or not via software control. Both SCTRL_CERT and the bootloader have TLV/Attribute to enable debug port; they have the ability to enable "*CM33_NS_DBG_PORT*", "*CM33_S_DBG_PORT*", "*DSP_DBG_PORT*", "*WIFI_DBG_PORT*", and "*BT_DBG_PORT*". Please note Wi-Fi debug port and BT debug port also need the control mechanism residing in Wi-Fi/BT; in other words, to use Wi-Fi debug port, you need to enable WIFI DBG port on both CM33 and Wi-Fi, and so does BT.

In addition, the MT793X supports JTAG password; the password can be appended as a TLV item in the bootloader or CERT file. If EFUSE_JTAG_HW_PWD_EN is blown, after the bootloader or certification file is verified, then BootROM parses the password, calculates the hash of password and feeds the hash to the SEJ

HW module, and the SEJ HW module compares the hash with the password HASH that is blown in EFUSE_JTAG_HW_PWD.

If SEJ reports the password checking is complete, it enables "*CM33_NS_DBG_PORT*", "*CM33_S_DBG_PORT*", and "*DSP_DBG_PORT*". Please note JTAG HW password does not enable Wi-Fi and BT debug ports, even if the password is correct.

Table 3-1 JTAG enable method comparison

Bootloader	SCTRL_CERT	JTAG hardware password
Can enable CM33_NS_DBG_PORT CM33_S_DBG_PORT DSP_DBG_PORT WIFI_DBG_PORT BT_DBG_PORT	Can enable CM33_NS_DBG_PORT CM33_S_DBG_PORT DSP_DBG_PORT WIFI_DBG_PORT BT_DBG_PORT	Can enable CM33_NS_DBG_PORT CM33_S_DBG_PORT DSP_DBG_PORT

Priority: SCTRL_CERT -> Bootloader -> JTAG hardware password

The MT793X supports JTAG control related eFuses for different purposes as shown in Table 3-2.

Table 3-2 JTAG related eFuse

Item	Description
EFUSE_SW_JTAG_CON	Debug port by default is off, and can be enabled by the software
EFUSE_JTAG_BY_TLV_DIS	Only SCTRL_CERT can enable debug port
EFUSE_JTAG_DIS	Disable JTAG permanently
EFUSE_JTAG_HW_PWD_EN	Enable JTAG hardware password feature
EFUSE_JTAG_HW_PWD	HASH of JTAG password

3.3 CERT File Layout

MT793X certification file layout is the same as the bootloader's, so they can be generated by image packing/signing tool; Figure 3-4 shows the file layout.

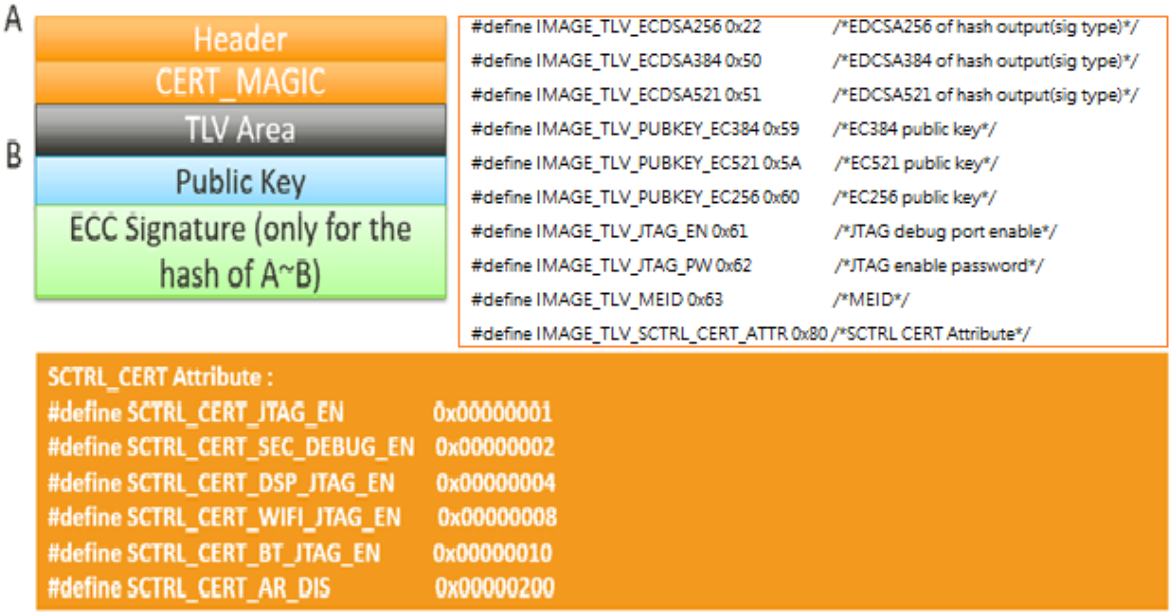

Figure 3-4 SCTRL_CERT file layout

4 Tool Security

4.1 Tool Authentication

BootROM does not support flash writer. But the software can be downloaded to SRAM for flash writing. This software is called Download Agent (DA).

MT793X download mode provides tool authentication to ensure the integrity and authentication of DA.

For tool authentication, with EFUSE_DAA_EN blown, PC tool needs to send authentication file first, and BootROM verifies the auth. file with SBC public key; if auth. file is verified, BootROM parses the TLV items in that.

As shown in Figure 4-1 and Figure 4-2, the authentication file carries the public key for DAA (DA authentication), HASH value of DA, and AUTH attribute that store the anti-rollback version and the enable bit to bind to DA

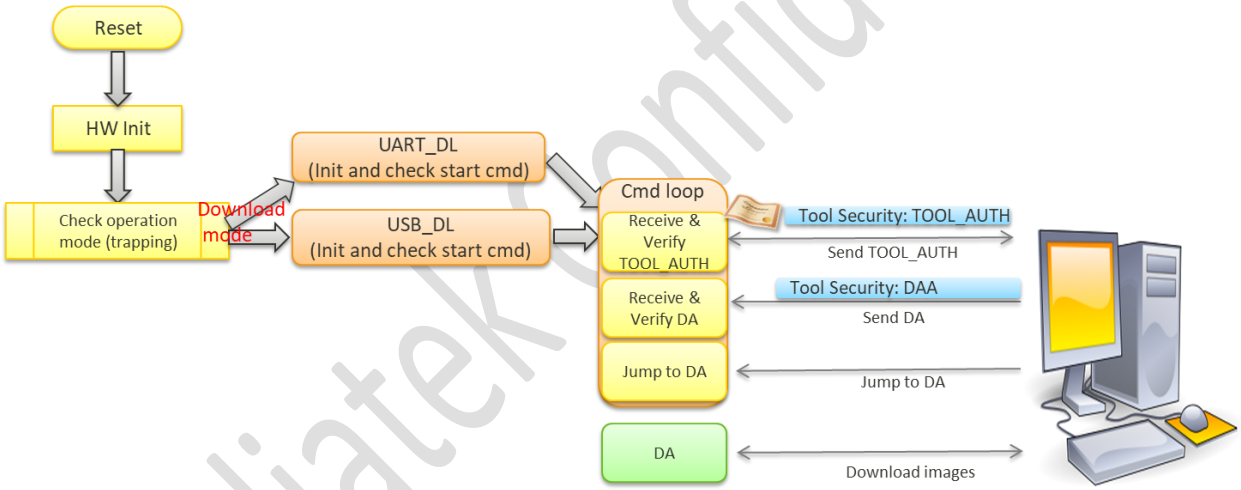

Figure 4-1 MT793X BROM secure download flow

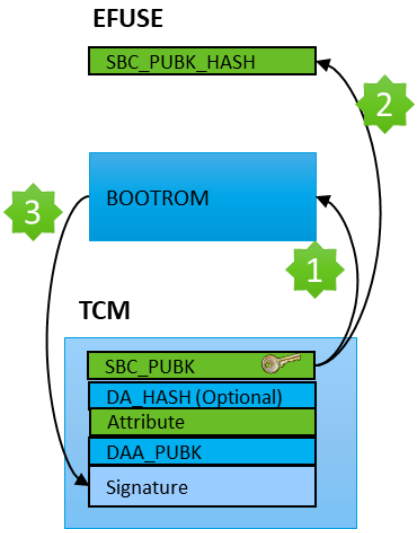

Figure 4-2 TOOL_AUTH verification flow

4.2 Tool Auth File Layout

The following figure shows that TOOL_AUTH is also checked for anti-rollback to prevent using the old DA.

Figure 4-3 Tool_Auth file layout

4.3 Download Agent Authentication (DAA)

If EFUSE_DAA_EN is blown, it means you need to verify the DA; the material to verify DA comes from TOOL_AUTH file that PC sends to the device in the previous step.

PC tool sends DA to the device in download mode. BootROM calculates the HASH of DA, and uses DAA public key to verify the signature of DA. If the signature is verified, and the attribute "TOOL_AUTH_BIND2DA" is enabled, BootROM compares the HASH value calculated by BootROM with what TOOL_AUTH is appended to.

As shown in Figure 4-4, first, BootROM calculates the HASH value of DA. Second, BootROM uses DAA_PUBK to verify signature and check whether the HASH value matches the value calculated by BootROM. If bind2da is set in TOOL_AUTH attribute, BootROM compares the DA_HASH with the one calculated by BootROM.

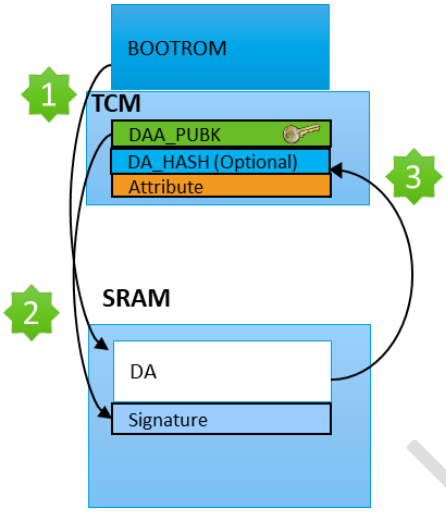

Figure 4-4 DA verification flow

To enable DAA feature, you must blow the eFuse in Table 4-1.

Table 4-1 DAA eFuse

Item	Description
EFUSE_DAA_EN	Enable DA authentication

5 Q & A

Q1. **Can XIP boot be disabled?**

A1. BootROM does not provide method to disable XIP boot. Only if the bootloader is verified, then bootROM can jump to the valid address specified in the bootloader header. Otherwise, BootROM may cause boot up failure. You can sign the RAM-boot bootloader image (with valid header) to make sure the bootloader can be executed in SRAM.

Q2. **Can the bootloader be loaded to PSRAM instead of SRAM?**

A2**. Unfortunately, NO**. Because it is necessary to calibrate the PSRAM before using it. BootROM is not suitable for this task and does not have enough ROM space to contain the calibration subroutine.

Q3. **Can low power sleep or wake up operation be performed without ROM reset?**

A3. The MT793x provides configure registers for the software to jump to any address after reset / wake up from deep sleep. If ROM reset is not allowed, the software can configure it to bypass ROM resume flow

Q4. **How big is the size of data processed by the AES-OTF per plaintext/ciphertext translation?**

A4. 64 bits (8 bytes) per plaintext/ciphertext translation.

Q5. **How does lockdown bits controlled by the software at run time?**

A5. You can use eFuse-API to change the lockdown bits (lockdown only, not recover) dynamically, by the software.

Exhibit 1 Terms and Conditions

Your access to and use of this document and the information contained herein (collectively this "Document") is subject to your (including the corporation or other legal entity you represent, collectively "You") acceptance of the terms and conditions set forth below ("T&C"). By using, accessing or downloading this Document, You are accepting the T&C and agree to be bound by the T&C. If You don't agree to the T&C, You may not use this Document and shall immediately destroy any copy thereof.

This Document contains information that is confidential and proprietary to MediaTek Inc. and/or its affiliates (collectively "MediaTek") or its licensors and is provided solely for Your internal use with MediaTek's chipset(s) described in this Document and shall not be used for any other purposes (including but not limited to identifying or providing evidence to support any potential patent infringement claim against MediaTek or any of MediaTek's suppliers and/or direct or indirect customers). Unauthorized use or disclosure of the information contained herein is prohibited. You agree to indemnify MediaTek for any loss or damages suffered by MediaTek for Your unauthorized use or disclosure of this Document, in whole or in part.

MediaTek and its licensors retain titles and all ownership rights in and to this Document and no license (express or implied, by estoppels or otherwise) to any intellectual propriety rights is granted hereunder. This Document is subject to change without further notification. MediaTek does not assume any responsibility arising out of or in connection with any use of, or reliance on, this Document, and specifically disclaims any and all liability, including, without limitation, consequential or incidental damages.

THIS DOCUMENT AND ANY OTHER MATERIALS OR TECHNICAL SUPPORT PROVIDED BY MEDIATEK IN CONNECTION WITH THIS DOCUMENT, IF ANY, ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE. MEDIATEK SPECIFICALLY DISCLAIMS ALL WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, FITNESS FOR A PARTICULAR PURPOSE, COMPLETENESS OR ACCURACY AND ALL WARRANTIES ARISING OUT OF TRADE USAGE OR OUT OF A COURSE OF DEALING OR COURSE OF PERFORMANCE. MEDIATEK SHALL NOT BE RESPONSIBLE FOR ANY MEDIATEK DELIVERABLES MADE TO MEET YOUR SPECIFICATIONS OR TO CONFORM TO A PARTICULAR STANDARD OR OPEN FORUM.

Without limiting the generality of the foregoing, MediaTek makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does MediaTek assume any liability arising out of the application or use of any product, circuit or software. You agree that You are solely responsible for the designing, validating and testing Your product incorporating MediaTek's product and ensure such product meets applicable standards and any safety, security or other requirements.

The above T&C and all acts in connection with the T&C or this Document shall be governed, construed and interpreted in accordance with the laws of Taiwan, without giving effect to the principles of conflicts of law.