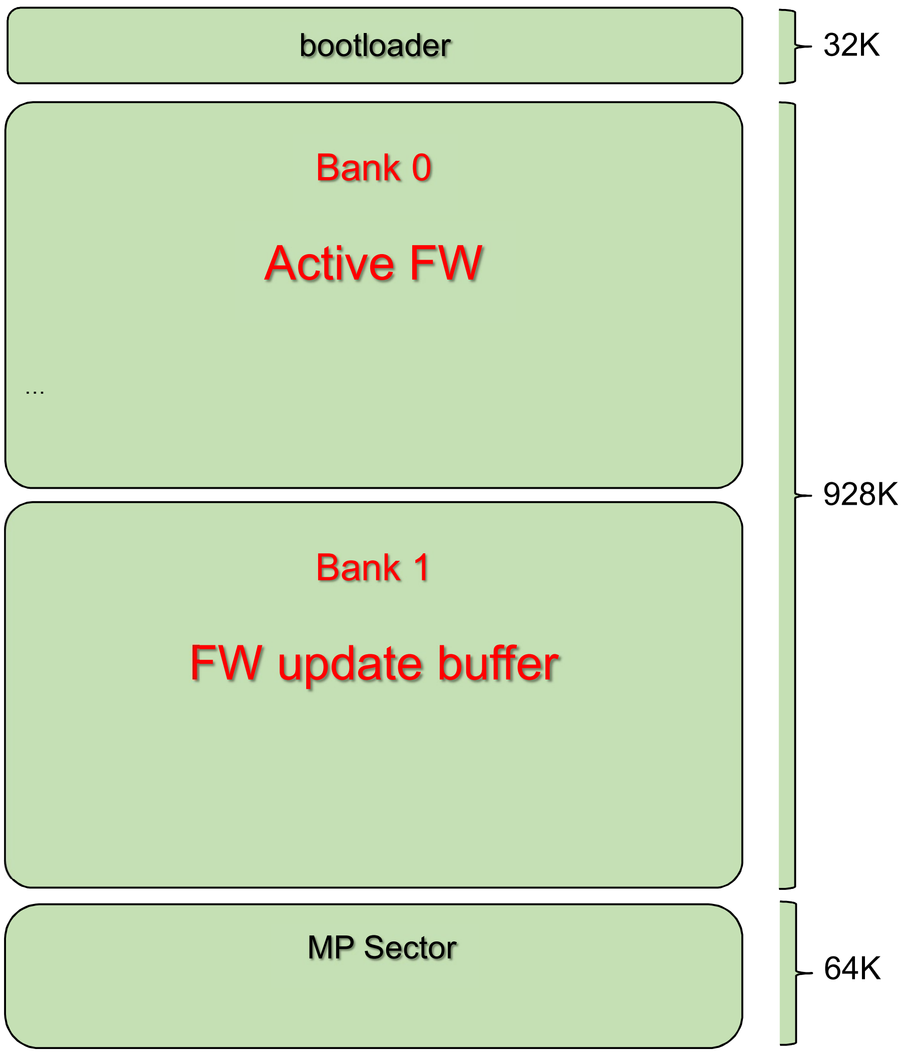
Rafael RT58x BLE SDK

OTA Guide V1.1

Table of Contents

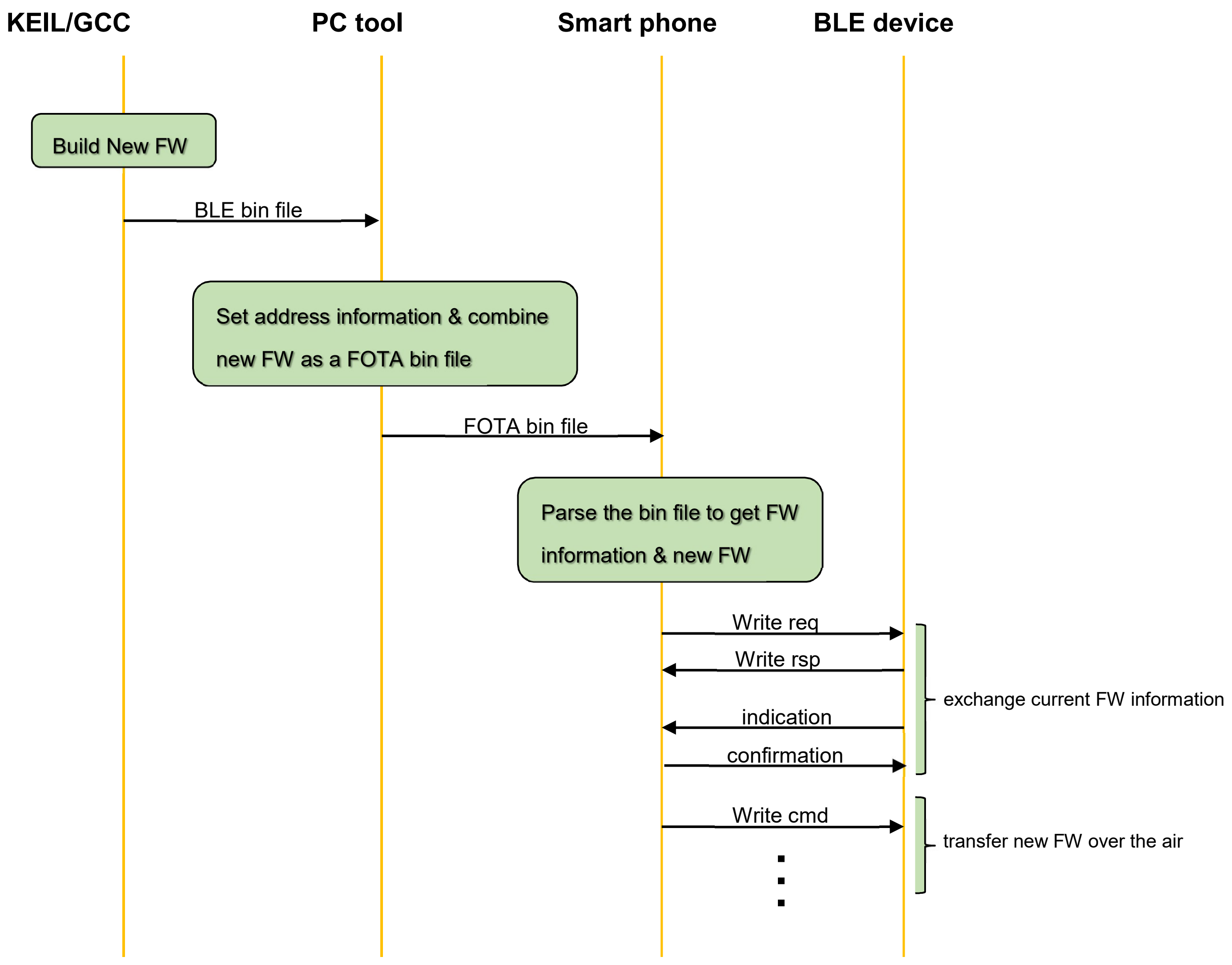
1. Overview

RT58x BLE SDK utilizes dual-bank flash to perform firmware over-the-air (FOTA) updates to ensure a valid image is activated. Bank 0 of the dual-bank flash is configured as an active FW block and Bank 1 is configured as the FW update buffer.



A complete FOTA update is accomplished via the following steps:

- Build new FW using KEIL/GCC.
- Use the PC tool to combine new FW and system information into a FOTA bin file.
- Load the FOTA bin file into smart phone and use the FOTA APP to parse the system information and FW.
- Query system information of the BLE device and make a decision whether or not to update the FW.


KEIL/GCC
PC tool
Smart phone
BLE device
Build New FW
BLE bin file
Set address information & combine new FW as a FOTA bin file
FOTA bin file
Parse the bin file to get FW information & new FW
Write req
Write rsp
indication
confirmation
exchange current FW information
Write cmd
transfer new FW over the air

2. Running FOTA Update Demo

Confirm BLE device support the functionality of FOTA update, which means the bootloader & FOTA update service already programmed

↓

Build new **BLE bin** file with new BLE system information (ex: *FW version*)

↓

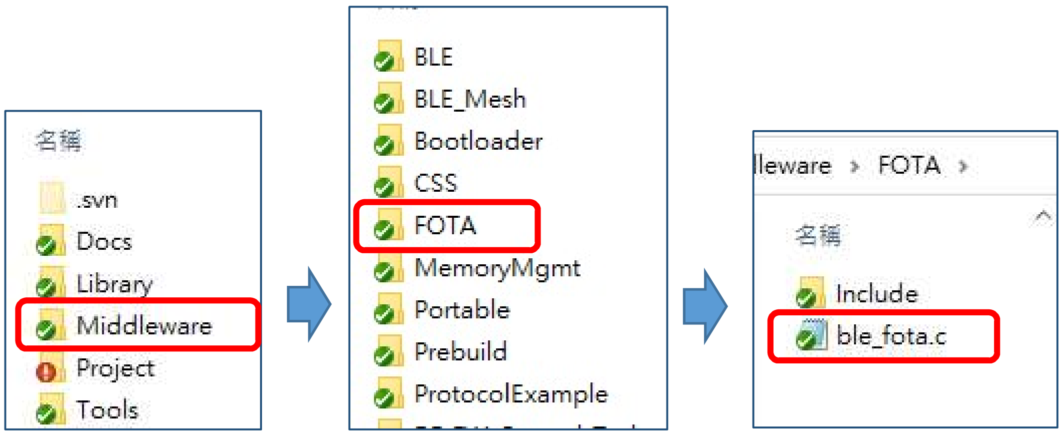
Load new **BLE bin** file into the PC tool and key-in the address information. The PC tool will combine the info as a **FOTA bin** file

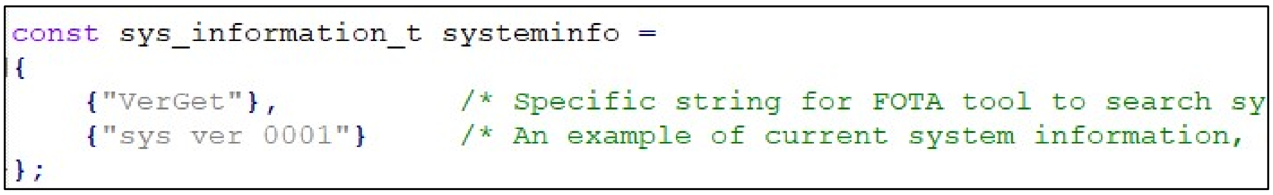
↓

Use mobile app to query the current *FW version* of the BLE device and decode the *FW version* of the **FOTA bin** file to determine whether or not to perform an over-the-air update process

2.1 Build the BLE bin file with BLE System information

BLE System information is set by array systeminfo [] which is defined in “ble_fota.c” file with 16-byte maximum length.



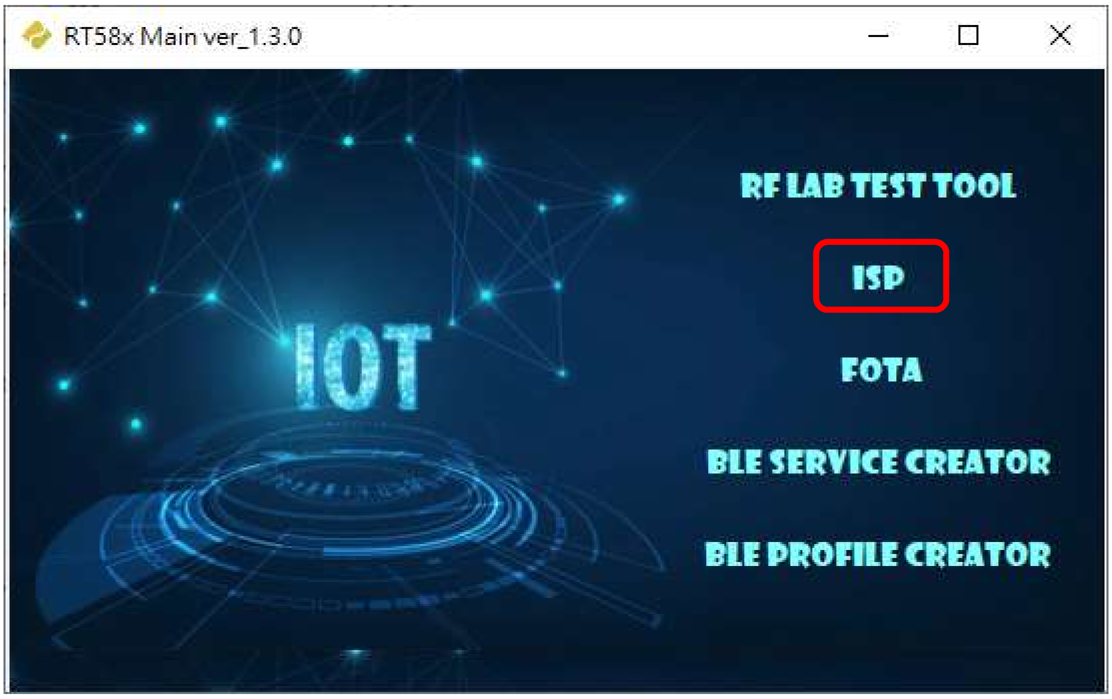


```
const sys_information_t systeminfo =
{
    {"VerGet"},         /* Specific string for FOTA tool to search sy
    {"sys ver 0001"}    /* An example of current system information,
};
```

The string “VerGet” is defined for PC tool to search system information, in the code base should not appear second string like this.

2.2 Using the IOT Evaluation Tool to Get FOTA bin File

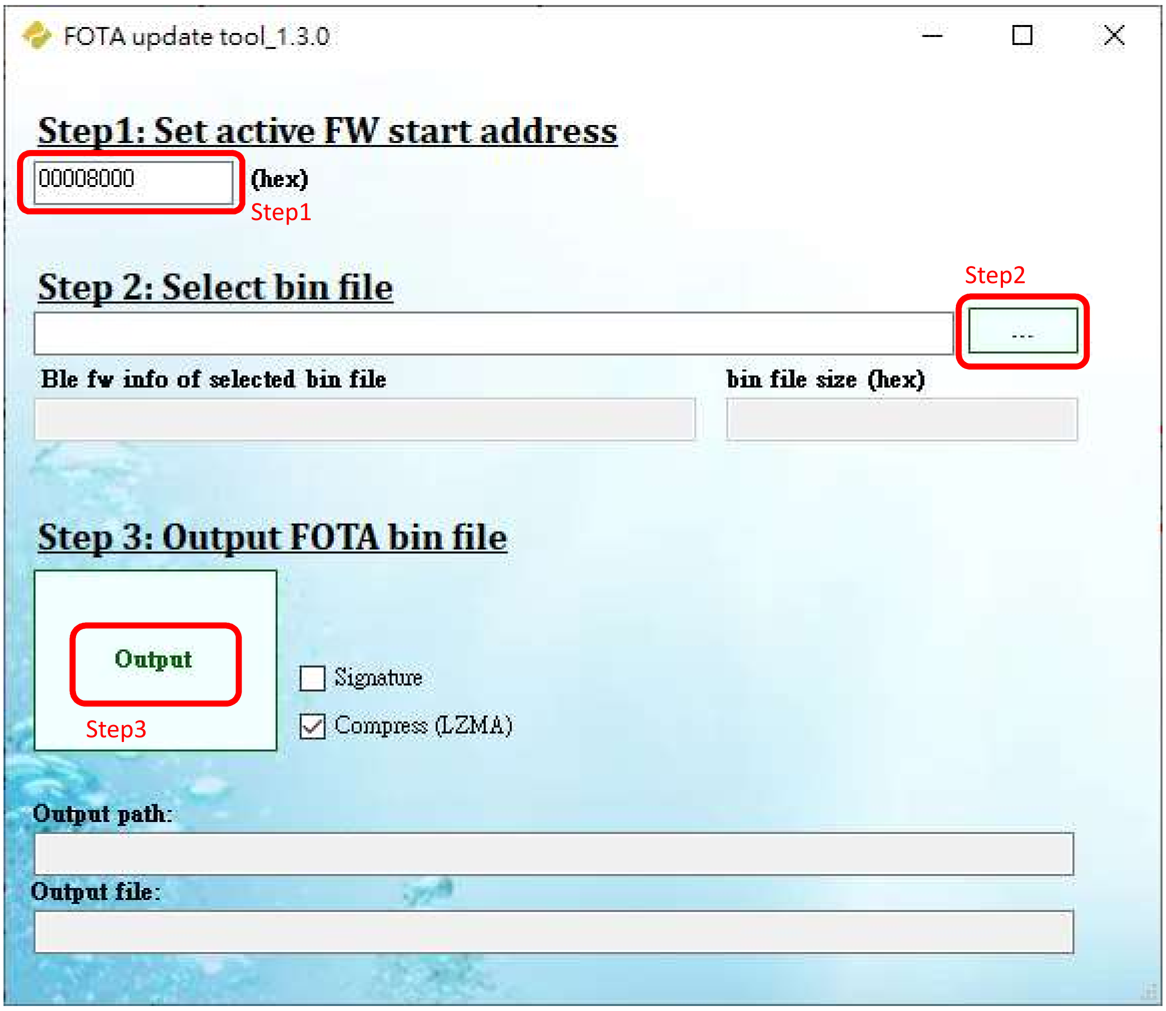
After open the IOT evaluation tool, click the icon “FOTA” to go to the page “FOTA update tool”.



In FOTA update tool, follows these steps to get FOTA bin file

2.2.1 Normal Mode

Step 1: Set active FW start address, default value is 0x8000, Usually, user don't have to change the value.
Step 2: Select the bin file that already built by KEIL/GCC.
Step 3: Click the "Output" button to output a new FOTA bin file.
[Note that after SDK version 1.3.0, it is recommended to check the "Compress(LZMA)" option.]



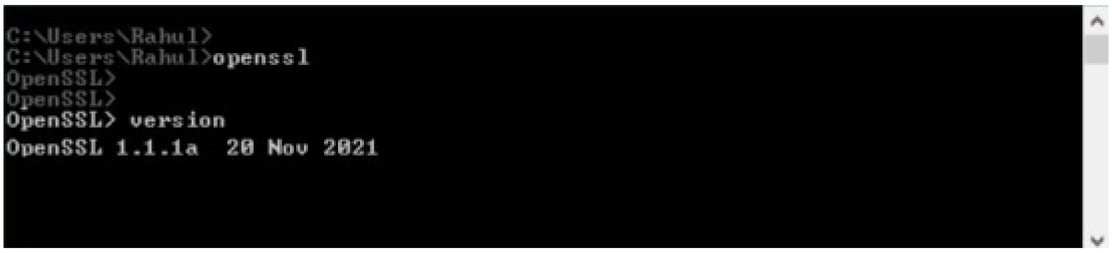
Signature: ☐
Compress (LZMA): ☑

2.2.2 Security Boot Mode

OpenSSL is an open-source command line tool that is commonly used to generate private keys, create CSRs, install your SSL/TLS certificate, and identify certificate information. Security Boot option is a quick way to implement these function with openssl.exe. This user guide will help you understand the most common OpenSSL commands and how to use them. Security Boot option is created with c sharp and will generate ECDSA digital signature and

verify signature by this software, the following third-party packages are required:
https://www.microsoft.com/en-us/download/confirmation.aspx?id=30679

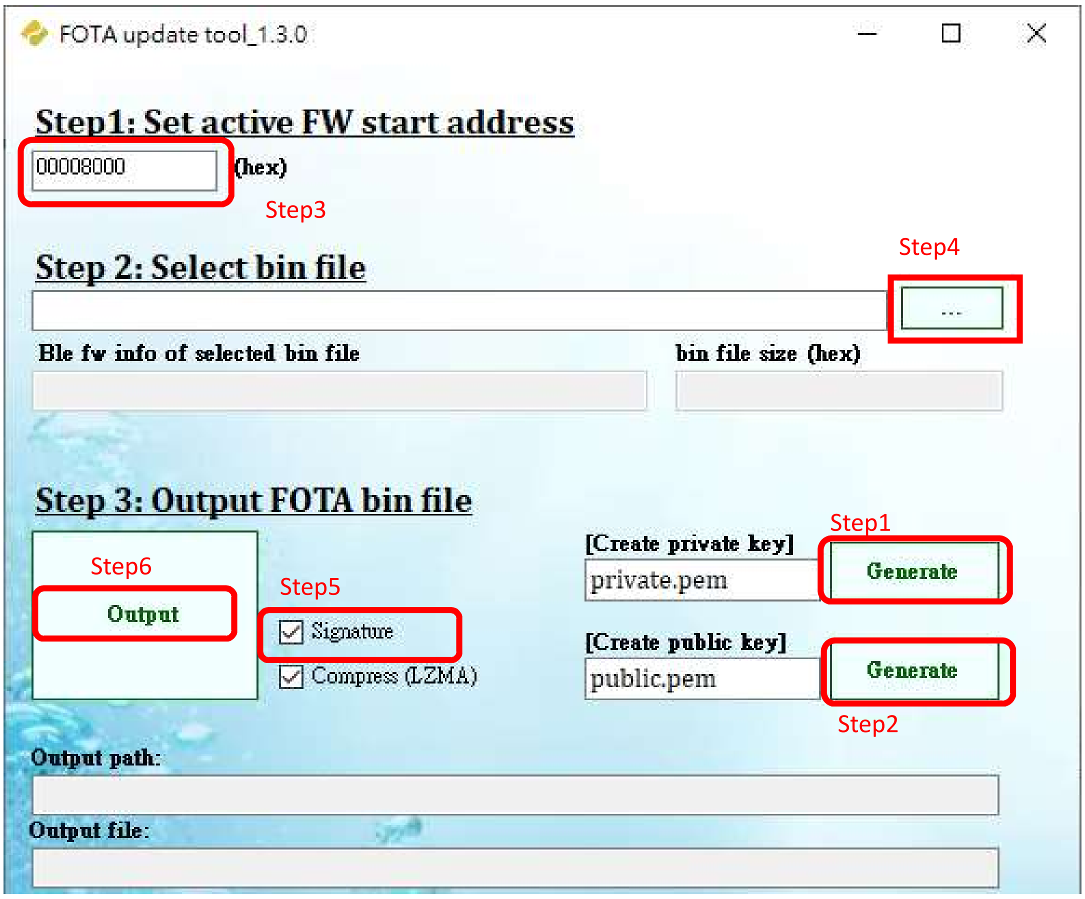
- Download OpenSSL Binary
  - i. Download the latest OpenSSL windows installer file. Click the below link to visit OpenSSL download page:
    http://slproweb.com/products/Win32OpenSSL.html
  - ii. Open a command prompt on your system and type "openssl" to open OpenSSL prompt. After that type version to get the installed OpenSSL version on your system (as shown in Figure 2.1).



```
C:\Users\Rahul>
C:\Users\Rahul>openssl
OpenSSL>
OpenSSL>
OpenSSL> version
OpenSSL 1.1.1a  20 Nov 2021
```

Figure 2.1 OpenSSL version

- Using SHA-256 Algorithm with ECDSA
  - i. Elliptic Curve Digital Signature Algorithm, or ECDSA, is one of the more complex public key cryptography encryption algorithms. ECDSA is used for asymmetric encryption and decryption, as well as for signing and verifying. ECDSA provides the same level of security as RSA but it does so while using much shorter key lengths. Therefore, for longer keys, ECDSA will take considerably more time to crack through brute-forcing attacks.
  - ii. Advantages of using ECDSA to RSA:
    - ECDSA requires much shorter keys to provide the same level of security.
    - ECDSA has quick process of signing and verification.
    - ECDSA has much better performance compared to RSA by shorter key lengths
    - ECDSA is in compliance with the modern requirements of industry.
- Tool architecture


FOTA update tool_1.3.0

Step1: Set active FW start address

00008000 (hex)

Step 2: Select bin file

Ble fw info of selected bin file

bin file size (hex)

Step 3: Output FOTA bin file

[Create private key] private.pem

[Create public key] public.pem

☑ Signature

☑ Compress (LZMA)

Output path:

Output file:

Figure 2.2 Security Boot Tool architecture

i. Figure 2.2 draws the Security Boot architecture diagram. Following step-by-step instructions, users can easily operate Security Boot.

ii. Generate Public and Private Keys with openssl.exe.

Step 1: Type in private key filename such as “private.pem” and click “GENERATE” button to generate private key.

Step 2: Proceed to the next step, type in public key filename such as “public.pem” and click “GENERATE” button to generate public key. Meanwhile, tool also will generate an output file named “public.der”, which is a public key certificate in .der format with OpenSSL.

iii. Generate and verify a signature with openssl.exe.

Step 3: Set active FW start address, default value is 0x8000, Usually, user don’t have to change the value. When selecting “bootloader.bin”, this value will be ignored

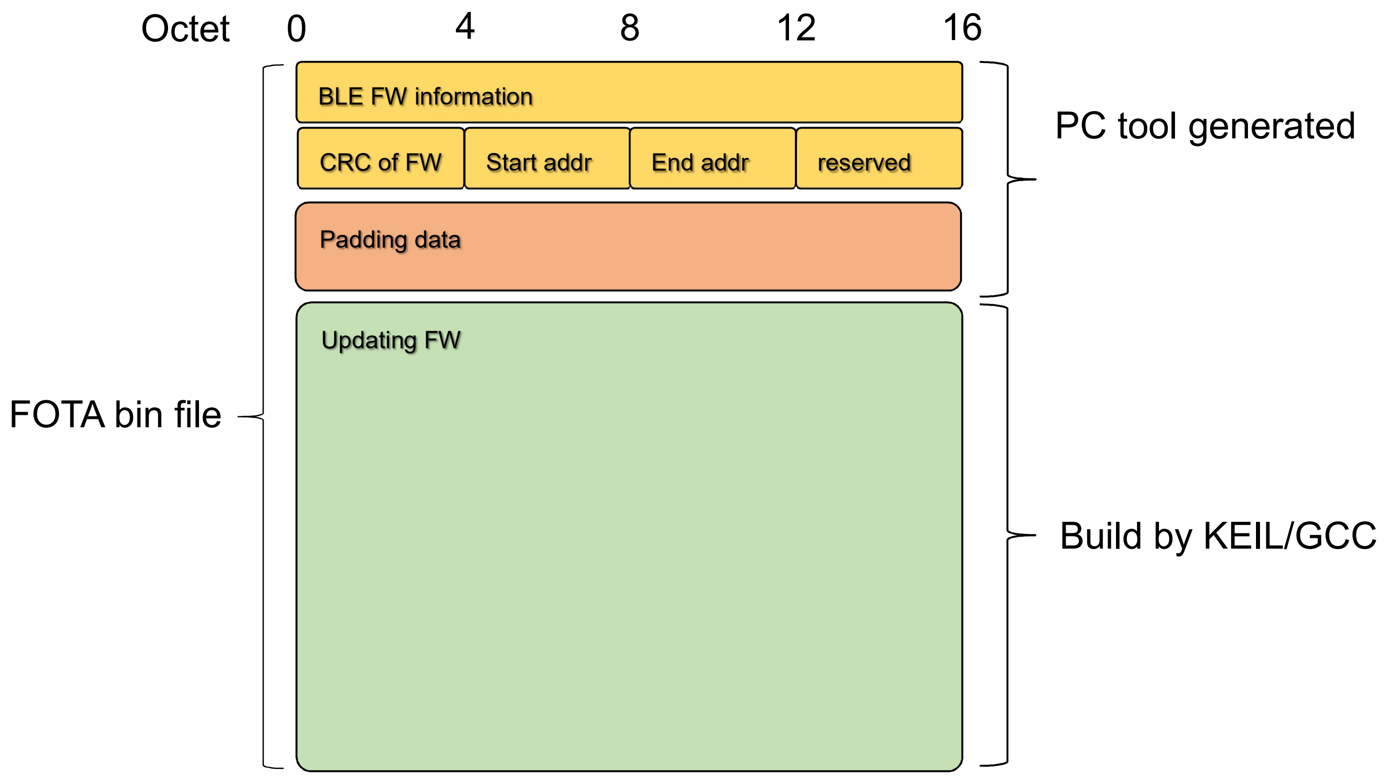
Step 4: Select digest of document to sign, such as “bootloader.bin”.

Step 5: Check the “signature” option.

Step 6: Pressing “Output” button, tool will parse the data of signature and write to end of file and output the file location information in "Output path" and "Output file".

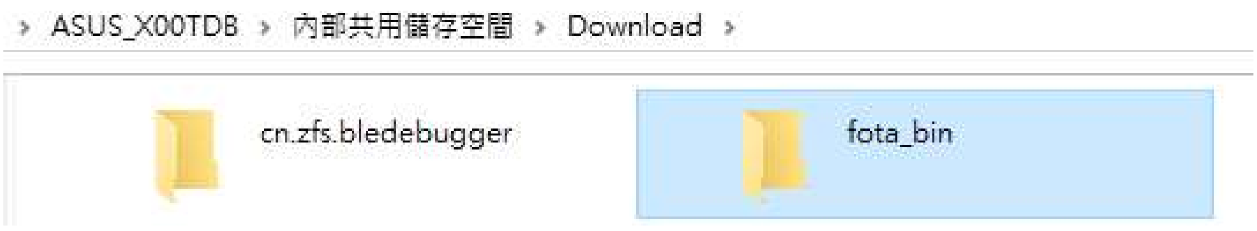
[Note that the “bootloader.bin” file cannot check the “Compress(LZMA)” option.]

PC tool generates 32byte of information. The information is saved to the front of FOTA bin file. PC tool also insert padding data behind the information. Padding data size equals to “active FW start address”, default is 0x8000. Updating FW is allocated after padding data.

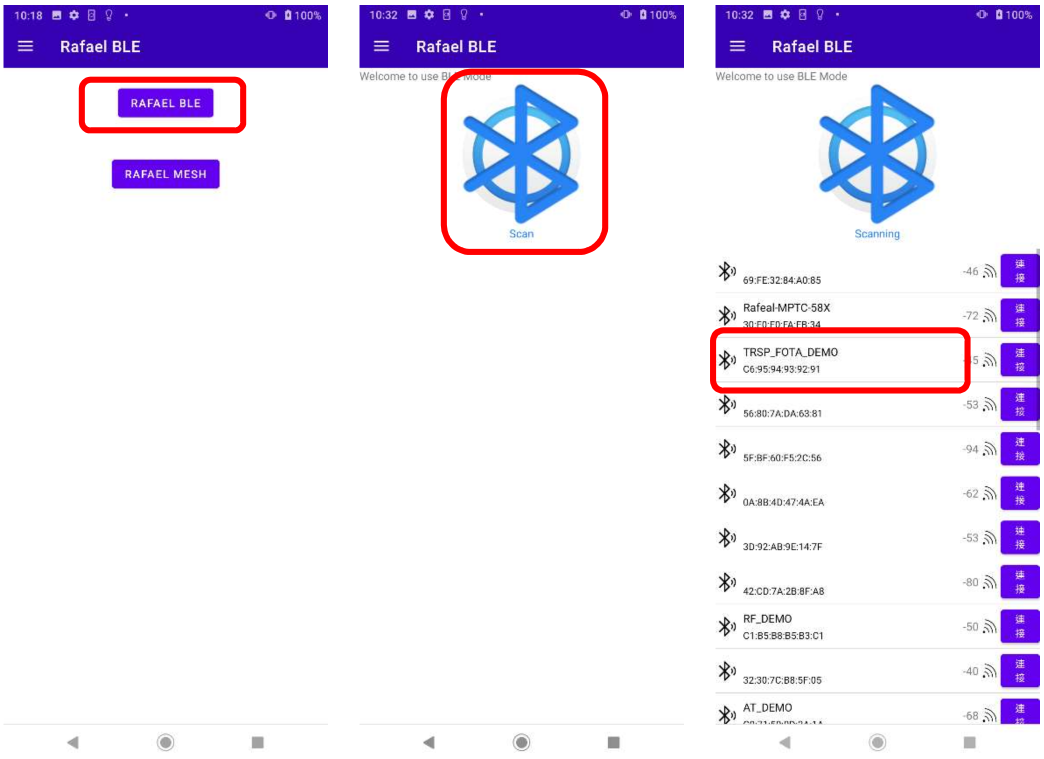


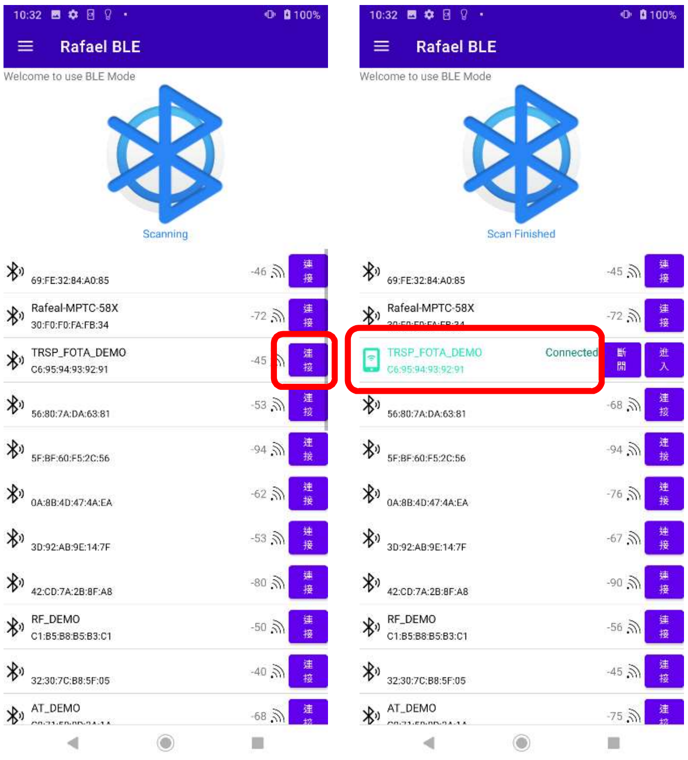
2.3 Start the FOTA Update by Using Mobile App

After generating the FOTA bin file, put it into specific folder “fota_bin” of smart phone (folder path: download/fota_bin).

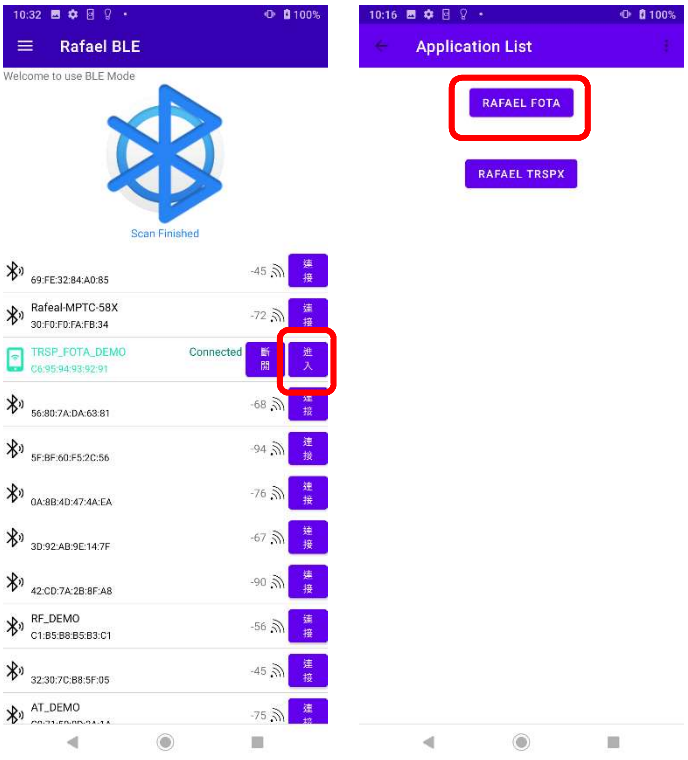


Install and open the mobile app “Rafael BLE” to scan and connect BLE device.


Rafael BLE
RAFAEL BLE
RAFAEL MESH
Welcome to use BLE Mode
Scan
Scanning
69:FE:32:84:A0:85 -46 連接
Rafeal-MPTC-58X 30:F0:F0:FA:FB:34 -72 連接
TRSP_FOTA_DEMO C6:95:94:93:92:91 連接
56:80:7A:DA:63:81 -53 連接
5F:BF:60:F5:2C:56 -94 連接
0A:8B:4D:47:4A:EA -62 連接
3D:92:AB:9E:14:7F -53 連接
42:CD:7A:2B:8F:A8 -80 連接
RF_DEMO C1:B5:B8:B5:B3:C1 -50 連接
32:30:7C:B8:5F:05 -40 連接
AT_DEMO -68 連接



Once BLE device successfully connect, enter application list to find FOTA update page and pairing with BLE device.



At FOTA update page,

Step1: select FOTA bin file by button SELECT BIN

Step2: click the CHECK VERSION button, mobile app will parse FOTA bin file to get system information and query BLE device's system information.

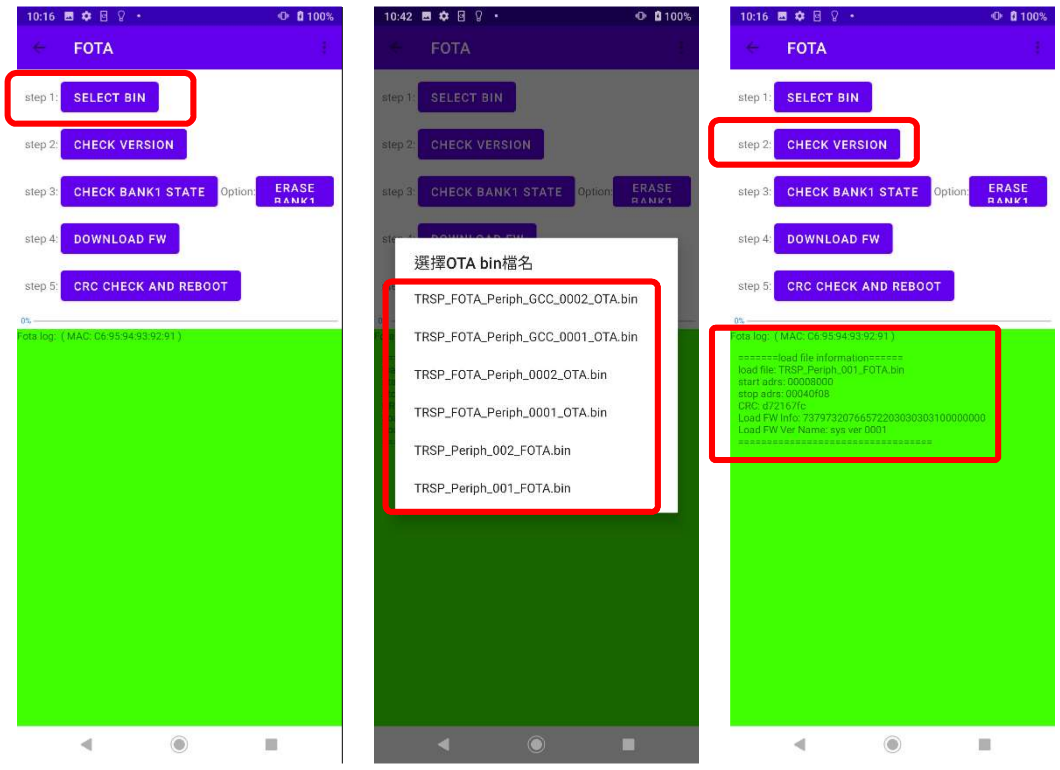
Step3: click the CHECK BANK1 STATE button to check updating FW buffer of device and APP the will check the transmission type is a resume one or new one.

※ Note that If previous incomplete update FW found in FW updating buffer, and FW mismatch with new updating FW. mobile APP will send erase command to device. Otherwise, mobile APP will resume the transmission from last abort address.

Step4: click button DOWNLOAD FW to start FW updating over the air.

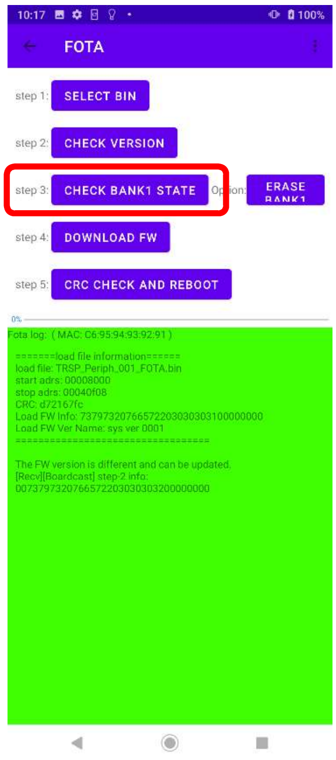
Step5: Signal device to update new FW and reboot.

Step 1 & 2:


10:16
100%
FOTA
step 1: SELECT BIN
step 2: CHECK VERSION
step 3: CHECK BANK1 STATE
Option: ERASE BANK1
step 4: DOWNLOAD FW
step 5: CRC CHECK AND REBOOT
0%
10:42
選擇OTA bin檔名
TRSP_FOTA_Periph_GCC_0002_OTA.bin
TRSP_FOTA_Periph_GCC_0001_OTA.bin
TRSP_FOTA_Periph_0002_OTA.bin
TRSP_FOTA_Periph_0001_OTA.bin
TRSP_Periph_002_FOTA.bin
TRSP_Periph_001_FOTA.bin

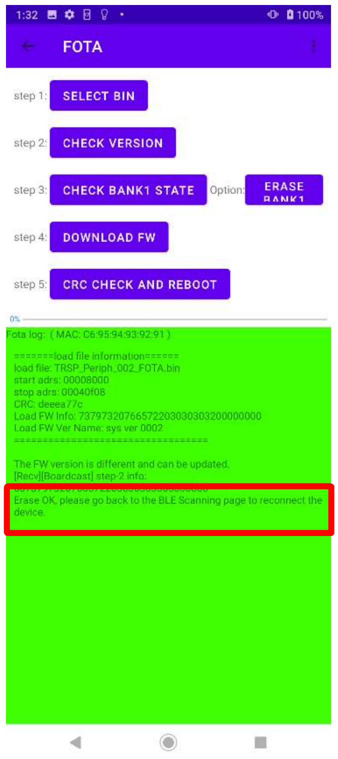
```
Fota log: ( MAC: C6:95:94:93:92:91 )
=======load file information=======
load file: TRSP_Periph_001_FOTA.bin
start adrs: 00008000
stop adrs: 00040f08
CRC: d72167fc
Load FW Info: 737973207665722030303031000000000
Load FW Ver Name: sys ver 0001
================================
```

Step 3 check FW buffer status



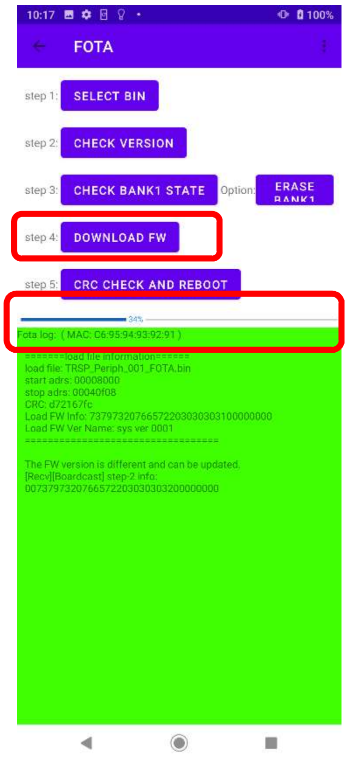
```
Fota log: ( MAC: C6:95:94:93:92:91 )
  =======load file information======
  load file: TRSP_Periph_001_FOTA.bin
  start adrs: 00008000
  stop adrs: 00040f08
  CRC: d72167fc
  Load FW Info: 7379732076657220303030310000000
  Load FW Ver Name: sys ver 0001
  ==================================

  The FW version is different and can be updated.
  [Recv][Boardcast] step-2 info:
  00737973207665722030303032000000000
```

Step 3-1 (optional) mobile APP will send erase command for the buffer to device, after erase command send, please go back to BLE scanning page to reconnect device and start from step 1

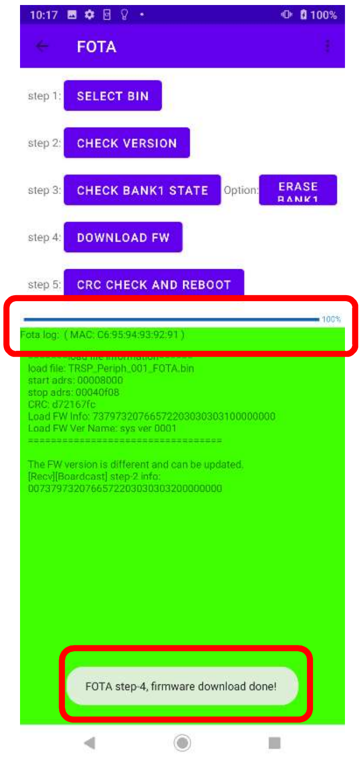


```
Fota log: ( MAC: C6:95:94:93:92:91 )
  =======load file information======
  load file: TRSP_Periph_002_FOTA.bin
  start adrs: 00008000
  stop adrs: 00040f08
  CRC: deeea77c
  Load FW Info: 7379732076657220303030320000000
  Load FW Ver Name: sys ver 0002
  ==================================

  The FW version is different and can be updated.
  [Recv][Boardcast] step-2 info:
  [illegible]
  Erase OK, please go back to the BLE Scanning page to reconnect the
  device.
```

Step 4: start send FW to device App shows the progress bar and log information.



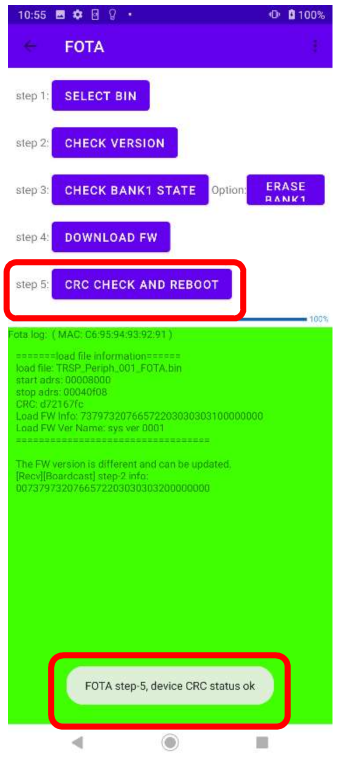
```
Fota log: ( MAC: C6:95:94:93:92:91 )
=======load file information======
load file: TRSP_Periph_001_FOTA.bin
start adrs: 00008000
stop adrs: 00040f08
CRC: d72167fc
Load FW Info: 7379732076657220303030310000000
Load FW Ver Name: sys ver 0001
==================================

The FW version is different and can be updated.
[Recv][Boardcast] step-2 info:
0073797320766572203030303200000000
```



```
Fota log: ( MAC: C6:95:94:93:92:91 )
=======load file information======
load file: TRSP_Periph_001_FOTA.bin
start adrs: 00008000
stop adrs: 00040f08
CRC: d72167fc
Load FW Info: 7379732076657220303030310000000
Load FW Ver Name: sys ver 0001
==================================

The FW version is different and can be updated.
[Recv][Boardcast] step-2 info:
0073797320766572203030303200000000
```

FOTA step-4, firmware download done!

Step 5: check device get FW completely and reboot device to update new FW

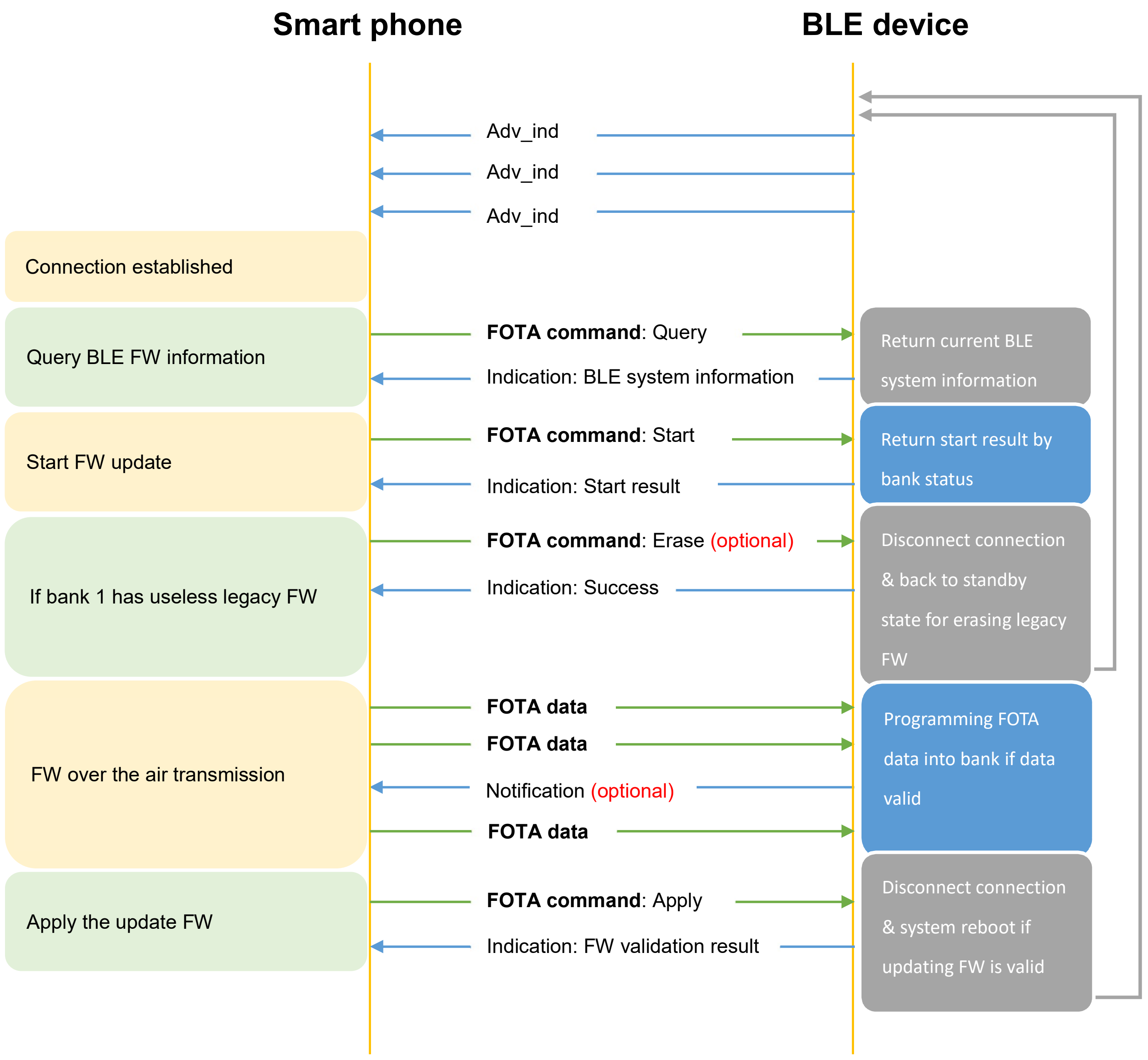


```
Fota log: ( MAC: C6:95:94:93:92:91 )
=======load file information======
load file: TRSP_Periph_001_FOTA.bin
start adrs: 00008000
stop adrs: 00040f08
CRC: d72167fc
Load FW Info: 7379732076657220303030310000000
Load FW Ver Name: sys ver 0001
==================================

The FW version is different and can be updated.
[Recv][Boardcast] step-2 info:
0073797320766572203030303200000000
```

FOTA step-5, device CRC status ok

3. FOTA Communication Flow

After the BLE device is paired with a smart phone, the compete FOTA update handshake flow proceeds as shown below:


Smart phone
BLE device
Adv_ind
Adv_ind
Adv_ind
Connection established
Query BLE FW information
FOTA command: Query
Indication: BLE system information
Return current BLE system information
Start FW update
FOTA command: Start
Indication: Start result
Return start result by bank status
If bank 1 has useless legacy FW
FOTA command: Erase (optional)
Indication: Success
Disconnect connection & back to standby state for erasing legacy FW
FW over the air transmission
FOTA data
FOTA data
Notification (optional)
FOTA data
Programming FOTA data into bank if data valid
Apply the update FW
FOTA command: Apply
Indication: FW validation result
Disconnect connection & system reboot if updating FW is valid

From the handshake flow, you can find that there were two major functions **FOTA command** and **FOTA data** to handle FOTA update procedure. These two functions defined in the file “ble_fota.c”:

```
void ble_fota_cmd(uint8_t host_id, uint8_t length, uint8_t *p_data);
void ble_fota_data(uint8_t host_id, uint8_t length, uint8_t *p_data);
```

3.1 FOTA command

FOTA commands were parsed by function "ble_fota_cmd()".

```
void ble_fota_cmd(uint8_t host_id, uint8_t length, uint8_t *p_data);
```

The first byte of command payload contains command ID and each command ID may contain different information behind. We defined four kind of commands here:

1. **Query**: Query device current system information.

octets	1
parameter	Command ID (0x00)

2. **Start**: Start FW upgrade, this command contains new FW length and CRC.

octets	1	4	4	4	1
parameter	Command ID (0x01)	FW length	FW CRC	Notify interval	FW info

3. **Erase**: Terminated the connection and erasing legacy FW and information.

octets	1
parameter	Command ID (0x02)

4. **Apply**: Apply the new FW if device receiving FW length and CRC matched with FOTA command "Start".

octets	1	4
parameter	Command ID (0x03)	Bank0 address

Note: command "Erase" sends only when device report that there are legacy FW in bank 1 and smart phone confirmed this FW was not match with updating FW. You can find the upon definitions in the file "ble_fota.h".

```
#define OTA_CMD_QUERY                                   0x00  /* Query current system information */
#define OTA_CMD_START                                   0x01  /* Start FW OTA update */
#define OTA_CMD_ERASE                                   0x02  /* Erase legacy FW */
#define OTA_CMD_APPLY                                   0x03  /* FW transmission completed, apply new FW */
```

When device received FOTA command, an indication with error code will return to smart phone to decide how to continue the FOTA procedure. These error codes were also defined in “ble_fota.h” file:

```
typedef uint8_t OtaErrCode;
#define OTA_ERR_CODE_NO_ERR                   0x00  /* Command success */
#define OTA_ERR_CODE_CMD_ERR                  0x01  /* Unsupported command ID  */
#define OTA_ERR_CODE_ALREADY_START            0x02  /* FOTA procedure already start */
#define OTA_ERR_CODE_UPDATE_NOT_START         0x03  /* FOTA procedure was not start */
#define OTA_ERR_CODE_FLASH_ERASE_ERR          0x04  /* Flash erase fail */
#define OTA_ERR_CODE_FW_CRC_ERR               0x05  /* Receiving FW's CRC incorrect */
#define OTA_ERR_CODE_FW_LEN_ERR               0x06  /* Receiving FW's length incorrect */
#define OTA_ERR_CODE_OUT_OF_BANK_SIZE         0x07  /* Updating FW's length larger than bank size */
```

1. **No error**: Command success.

octets	1
parameter	Error code (0x00)

2. **Command error**: Unsupported command ID.

octets	1
parameter	Error code (0x01)

3. **Already start**: FOTA procedure already start.

octets	1	4	4	3
parameter	Error code (0x02)	Updating FW length	Updating FW CRC	Updating FW next expect address

4. **Update not start**: FOTA procedure not start yet.

octets	1
parameter	Error code (0x03)

5. **Flash erase error**: Bank flash erase fail.

octets	1
parameter	Error code (0x04)

6. **FW CRC error**: Receiving FW's CRC is incorrect.

octets	1

parameter	Error code (0x05)

7. **FW length error**: Receiving FW's length is incorrect.

octets	1
parameter	Error code (0x06)

8. **Out of bank size**: Updating FW's length larger than bank size.

octets	1	3
parameter	Error code (0x07)	Current FW bank size

3.2 FOTA data

The function “ble_fota_data()” is implemented to handle FOTA data.

```
void ble_fota_data(uint8_t host_id, uint8_t length, uint8_t *p_data);
```

First 4 bytes of data payload is data header which contains the FOTA data programming address (3 bytes) and length (1byte).

octets	3	1	Variable
parameter	Data programming address	Data programming length	Data

If there were invalid data header, device will send notification to response it. Hence, for ideal FOTA data transmission, there were not any notification send from device unless smart phone had configure periodic notification. The notification reason is contained in the first byte of notification packet and you can find the definition in the “ble_fota.h” file as follows:

```
/*Notification reasons for FOTA data*/
typedef uint8_t OtaNotify;
#define OTA_DATA_NOTIFY_PERIODIC                0x00  /* Periodic notification, the notify interva
#define OTA_DATA_NOTIFY_TIMEOUT                 0x01  /* FOTA data was not received in a specific
#define OTA_DATA_NOTIFY_ADDRESS_UNEXPECTED      0x02  /* Received FOTA data's address was not cont
#define OTA_DATA_NOTIFY_LEN_ERROR               0x03  /* Received FOTA data length incorrect */
#define OTA_DATA_NOTIFY_TOTAL_LEN_ERR           0x04  /* Total received FOTA data length is larger
#define OTA_DATA_NOTIFY_ADDRESS_ERR             0x05  /* Received FOTA data's address is larger th
#define OTA_DATA_NOTIFY_NOT_START               0x06  /* FOTA data received but FOTA procedure was
#define OTA_DATA_NOTIFY_NONE                    0xFF  /* No notification needs to send */
```

1. **Periodic**: Periodic notification, the notify interval set from FOTA start command.

octets	1	3
parameter	Notify reason (0x00)	Updating FW next expect address

2. **Timeout**: FOTA data was not received in a specific interval.

octets	1	3
parameter	Notify reason (0x01)	Updating FW next expect address

3. **Address unexpected**: Received FOTA data's address was not continuously.

octets	1	3
parameter	Notify reason (0x02)	Updating FW next expect address

4. **Length error**: Received FOTA data length incorrect.

octets	1	1
parameter	Notify reason (0x03)	Received FOTA data length

5. **Total length error**: Total received FOTA data length is larger than bank size.

octets	1	3
parameter	Notify reason (0x04)	Total Receiving FOTA data length

6. **Address error**: Received FOTA data's address is larger than bank size.

octets	1
parameter	Notify reason (0x05)

7. **Not start**: FOTA data received but FOTA procedure was not start.

octets	1
parameter	Notify reason (0x06)

Contact Information

Address	8F., No.28, Chenggong 12th St., Zhubei City, Hsinchu County 30264 Taiwan R.O.C.		
Contact	Tel: 886-3-5506258	Fax: 886-3-5506228	www.rafaelmicro.com
Sales	**World-wide**	886-3-5506258 ext. 206	michael.gauer@rafaelmicro.com
	China	86-1360-2679953	evan.tu@rafaelmicro.com
	Korea	82-10-5580-0657	hyunsu.lee@rafaelmicro.com
Technical Support		886-3-5506258 ext. 302	yenchih.shen@rafaelmicro.com

Revision History

Revision	Description	Owner	Date
1.0	Initial version	Nat	2022/2/9
1.1	Add Security Boot Mode description.	Ryan	2022/12/13