RTL8730E MP FLOW

Version 15.0

DISCLAIMER
Please Read Carefully: Realtek Semiconductor Corp., (Realtek) reserves the right to make corrections, enhancements, improvements and other changes to its products and services. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. Reproduction of significant portions in Realtek data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Realtek is not responsible or liable for such reproduced documentation. Information of third parties may be subject to additional restrictions. Buyers and others who are developing systems that incorporate Realtek products (collectively, “Customers”) understand and agree that Customers remain responsible for using their independent analysis, evaluation and judgment in designing their applications and that Customers have full and exclusive responsibility to assure the safety of Customers' applications and compliance of their applications (and of all Realtek products used in or for Customers’ applications) with all applicable regulations, laws and other applicable requirements. Designer represents that, with respect to their applications, Customer has all the necessary expertise to create and implement safeguards that (1) anticipate dangerous consequences of failures, (2) monitor failures and their consequences, and (3) lessen the likelihood of failures that might cause harm and take appropriate actions. Customer agrees that prior to using or distributing any applications that include Realtek products, Customer will thoroughly test such applications and the functionality of such Realtek products as used in such applications. Realtek’s provision of technical, application or other design advice, quality characterization, reliability data or other services or information, including, but not limited to, reference designs and materials relating to evaluation kits, (collectively, “Resources”) are intended to assist designers who are developing applications that incorporate Realtek products; by downloading, accessing or using Realtek’s Resources in any way, Customer (individually or, if Customer is acting on behalf of a company, Customer’s company) agrees to use any particular Realtek Resources solely for this purpose and subject to the terms of this Notice. Realtek’s provision of Realtek Resources does not expand or otherwise alter Realtek’s applicable published warranties or warranty disclaimers for Realtek’s products, and no additional obligations or liabilities arise from Realtek providing such Realtek Resources. Realtek reserves the right to make corrections, enhancements, improvements and other changes to its Realtek Resources. Realtek has not conducted any testing other than that specifically described in the published documentation for a particular Realtek Resource. Customer is authorized to use, copy and modify any individual Realtek Resource only in connection with the development of applications that include the Realtek product(s) identified in such Realtek Resource. No other license, express or implied, by estoppel or otherwise to any other Realtek intellectual property right, and no license to any technology or intellectual property right of Realtek or any third party is granted herein, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which Realtek products or services are used. Information regarding or referencing third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of Realtek Resources may require a license from a third party under the patents or other intellectual property of the third party, or a license from Realtek under the patents or other Realtek’s intellectual property. Realtek’s Resources are provided “as is” and with all faults. Realtek disclaims all other warranties or representations, express or implied, regarding resources or use thereof, including but not limited to accuracy or completeness, title, any epidemic failure warranty and any implied warranties of merchantability, fitness for a particular purpose, and non-infringement of any third party intellectual property rights. Realtek shall not be liable for and shall not defend or indemnify Customer against any claim, including but not limited to any infringement claim that related to or is based on any combination of products even if described in Realtek Resources or otherwise. In no event shall Realtek be liable for any actual, direct, special, collateral, indirect, punitive, incidental, consequential or exemplary damages in connection with or arising out of Realtek’s Resources or use thereof, and regardless of whether Realtek has been advised of the possibility of such damages. Realtek is not responsible for any failure to meet such industry standard requirements. Where Realtek specifically promotes products as facilitating functional safety or as compliant with industry functional safety standards, such products are intended to help enable customers to design and create their own applications that meet applicable functional safety standards and requirements. Using products in an application does not by itself establish any safety features in the application. Customers must ensure compliance with safety-related requirements and standards applicable to their applications. Designer may not use any Realtek products in life[1]critical medical equipment unless authorized officers of the parties have executed a special contract specifically governing such use. Life-critical medical equipment is medical equipment where failure of such equipment would cause serious bodily injury or death. Such equipment includes, without limitation, all medical devices identified by the U.S.FDA as Class III devices and equivalent classifications outside the U.S.
Customers agree that it has the necessary expertise to select the product with the appropriate qualification designation for their applications and that proper product selection is at Customers’ own risk. Customers are solely responsible for compliance with all legal and regulatory requirements in connection with such selection. Customer will fully indemnify Realtek and its representatives against any damages, costs, losses, and/or liabilities arising out of Designer’s non-compliance with the terms and provisions of this Notice.

TRADEMARKS
Realtek is a trademark of Realtek Semiconductor Corporation. Other names mentioned in this document are trademarks/registered trademarks of their respective owners.

USING THIS DOCUMENT
This document is intended for the software engineer’s reference and provides detailed programming information. Though every effort has been made to ensure that this document is current and accurate, more information may have become available subsequent to the production of this guide.

Revision History:

Revision	Date	Changes	Author
1.0	2022/11/15	First Release.	Winnita, Tim
2.0	2023/04/06	Delete UI sample code(3.1.1.2) Update Appendix 12.1 for EFuse contents.	Winnita, Tim
3.0	2023/04/11	File format changes.	Winnita, Tim
4.0	2023/04/19	Merge bt chip number.	Winnita, Tim
5.0	2023/05/04	Add rfe_type setting and ant_select cmd for WLAN.	Winnita, Tim
6.0	2023/05/23	Add 3.1.3.7 Write and read fake Efuse.	Winnita, Tim
7.0	2023/07/28	Modify 6./12.Efuse content	Winnita, Tim
8.0	2023/10/18	Add s1 information	Winnita, Tim
9.0	2023/11/07	Modify 6./12.Efuse content	Winnita, Tim
10.0	2023/12/04	Modify WiFi 2G target power	Winnita, Tim
11.0	2024/03/29	Modify description(Efuse to OTP)	Winnita, Tim
12.0	2024/05/20	Modify Appendix/OTP_map for country code setup.	Winnita, Tim
13.0	2024/09/30	Modify BT OTP default value. Add 2-antenna TRX performance verification.	Winnita, Tim
14.0	2025/01/20	Add aux path RSSI calibration.	Winnita, Tim
15.0	2025/03/20	1. Modify 3.1.3.4 TSSI DE calculation example 2. Add “iwpriv mp_pwrctldm stop” command before calibration 3. Modify Table 7 Rx gain calibration channels 4. Modify Table 10 Rx gain verification channels	Winnita, Tim

Table of Contents

List of Tables

1 Test platform

The calibration flow described in following sections are based on WiFi/Bluetooth tester. Realtek had qualified tester is listed below:

Included 802.11ax Test		**Included 802.11ac Test**		**Only 802.11a/b/g/n Test**	
Vendor	Modal Name	Vendor	Modal Name	Vendor	Modal Name
LitePoint	IQxel-MW	LitePoint	IQxel	LitePoint	IQFlex
Itest	WT-328	Itest	WT-200	LitePoint	IQView
		NI	PXIe-5644R/5645R	LitePoint	IQnxn
		Anritsu	MT8870A	Agilent	N4010A
		Aeroflex	PXI 3000 Series		

The test environment setup is as below:

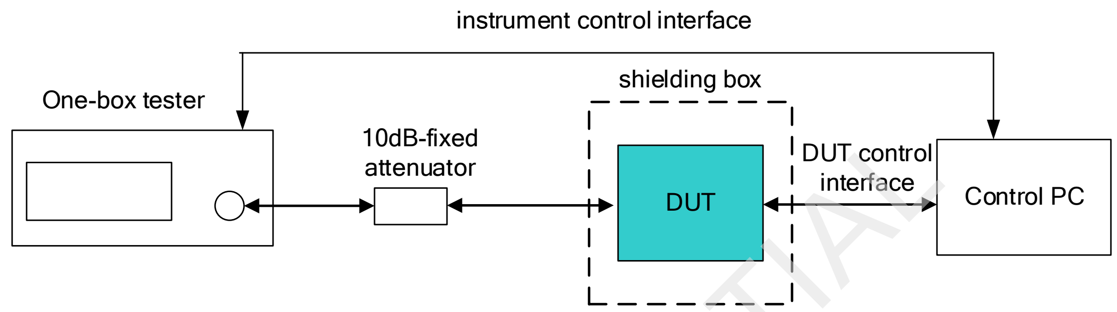


Figure 1: Brief diagram about test environment setup of 1-antenna DUT

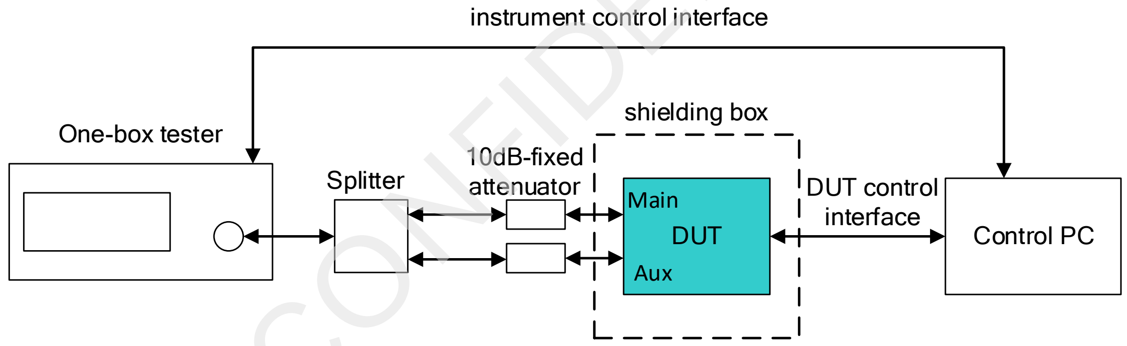


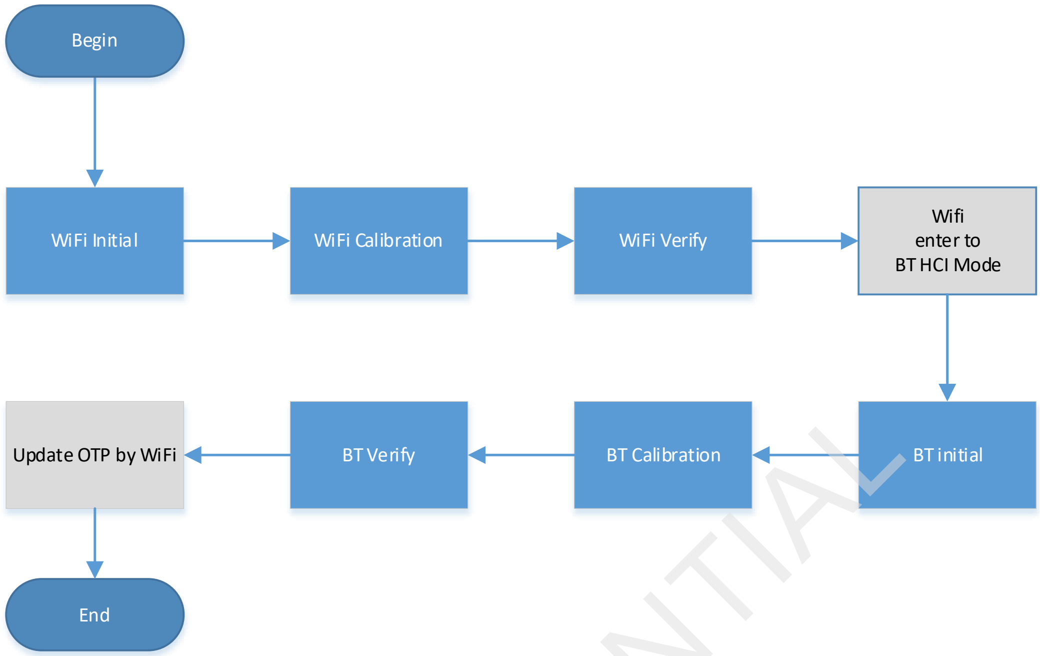
Figure 2: Brief diagram about test environment setup of 2-antenna DUT

NOTE

To reduce the mismatch effect between DUT and environment, 10dB-fixed attenuator is supposed to set closed to DUT as possible.

2 DUT MP Flow

The below diagram shows a global view of Mass-Production-Flow for testing WLAN and BT, please refer to the following sub-section to get the detail description of each step.



3 WiFi Mass Production Flow

3.1 WiFi calibration

The WiFi calibration steps are shown as Figure 2:

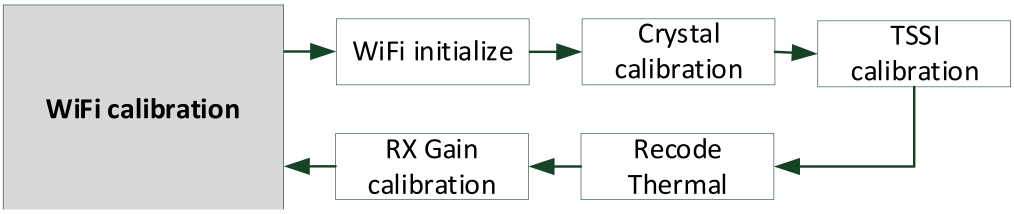


Figure 3: WiFi calibration flow

3.1.1 WiFi initialize

3.1.1.1 Environment Setup

First you need to check comport show up.
COM port number is based on your environment.

3.1.1.2 WiFi Initial Step

The relative commands of DUT WiFi enter or exit MP (Mass Production) is listed as below:

```
iwpriv mp_rfe 7          //Set rfe type, default value is 7.
```

NOTE

This command only works before mp_start. If the parameters are not given, the default value depend on the OTP value, and if the parameters and OTP value are not given, the default value is 7.

NOTE

Please refer to 'Ameba Smart RFE Type Suggestion' *for more details.*
If there is any concern, please contact Realtek FAE.

```
iwpriv mp_start
iwpriv mp_stop
```

```
#iwpriv mp_start

[MEM] After do cmd, available heap 69776
```

This command is recommended before calibration:

```
iwpriv mp_pwrctldm stop
```

The reason is that if the DUT has already been PG, the dynamic mechanism will be activated, this may lead to the calibration with unexpected results.

3.1.1.3 Pre-heating

Pre-heating is an one-time experiment to grab essential setup, it made DUT closed to the usual temperature and improved the stability of mass production. It will pre-heat the DUT for seconds before mass production.

Step 1 : Read and record thermal value THERRX at steady Rx status which lasts 5 minutes at room temperature.

```
iwpriv mp_rfe 7          //Set rfe type, default value is 7.
iwpriv mp_start          //Enter WiFi MP mode
iwpriv mp_antsw 1        //Select 1/2 TRX antenna(according to rfe_type),default is 1.
iwpriv mp_ther           //Get the thermal value THERinit
```

Record the THERinit value.
Wait for 5 minute.

```
iwpriv mp_ther           //Get the thermal value THERRX
```

Record the THERRX value.

NOTE

If THERRX – THERinit ≦ 3, there is no need to do the Pre-heating, Please skip the following steps and go to the next section.

Step 2 : Use MP HW Tx to do the following pre-heating operation.

```
iwpriv mp_channel 6
iwpriv mp_pwrtrk 1
iwpriv mp_txpower dbm=17
iwpriv mp_hwtx period=100,len=1500,count=10
                    //Supposed value of TxPacketInterval = 100
                    //Supposed initial value of TxPacketLength = 1500
iwpriv mp_ther      //Get the heated thermal value THERHeating
```

Step 3 : To figure out pre-heat settings (TxPacketInterval/TxPacketLength/TIMEHeating).

Trim the TxPacketInterval or TxPacketLength, then do HW Tx and record how long does the DUT be heated to THERRX temperature. If **THERHeating - THERRx ≦ 3**, means DUT is heated to desired temperature. The time-spent records as **TIMEHeating**.

Use these settings (**TxPacketInterval/TxPacketLength/TIMEHeating**) as pre-heating setup.

Pre-heating DUT before calibration by the found settings set:

Do HW Tx

```
iwpriv mp_hwtx period= TxPacketInterval,len= TxPacketLength,count=0
```

Last for TIMEHeating. So that DUT can be pre-heated to desired thermal range.

Example :

```
iwpriv mp_rfe 7         //Set rfe type, default value is 7.
iwpriv mp_start         //Enter WiFi MP mode
```

Wait for 5 minute.

```
iwpriv mp_ther           //MP feedback the thermal value = 2d
```

Record the THERRX = 2d

```
iwpriv mp_channel 6
iwpriv mp_pwrtrk 1
iwpriv mp_txpower dbm=17
iwpriv mp_hwtx period=200,len=1500,count=0
```

Example:
Use HW Tx, HT20 HTMCS7 channel 6, TxPacketInterval = 200, TxPacketLength = 1500 to pre-heat DUT.
5 seconds past.

```
iwpriv mp_ther           //MP feedback the thermal value = 2e
```

Record the THERHeating = 2e

2e (THERHeating) – 2d (THERRX) ≦ 3,DUT is heated to desired thermal range!

(5 seconds is just a sample value here. In the operation, please do HW Tx and observe the THERHeating variance. How long does the DUT be heated to the thermal range.)

Finally, get a pre-heating settings set :

```
iwpriv mp_hwtx period=200,len=1500,count=0
```

Heating time-spent is TIMEHeating = 5 seconds.

DUT can be heated to desired thermal rang by
HW Tx(iwpriv mp_hwtx period=200,len=1500,count=0) for 5 seconds (TIMEHeating).

```
iwpriv mp_hwtx stop         //Stop HW Tx
```

The pre-heating settings were found.
Please pre-heat DUT before calibration.

3.1.2 Crystal Calibration

3.1.2.1 OTP definition about Crystal Cap

First, take a look at calibration data content about setting of Crystal Cap. Normal driver will load this value in initial step. So this value must be well-calibrated and filled on correct calibration data location.

Crystal Calibration	**0xC9[7:0]**

Table 1: Crystal calibration offset in OTP

3.1.2.2 Calibrated Crystal Cap. Flow

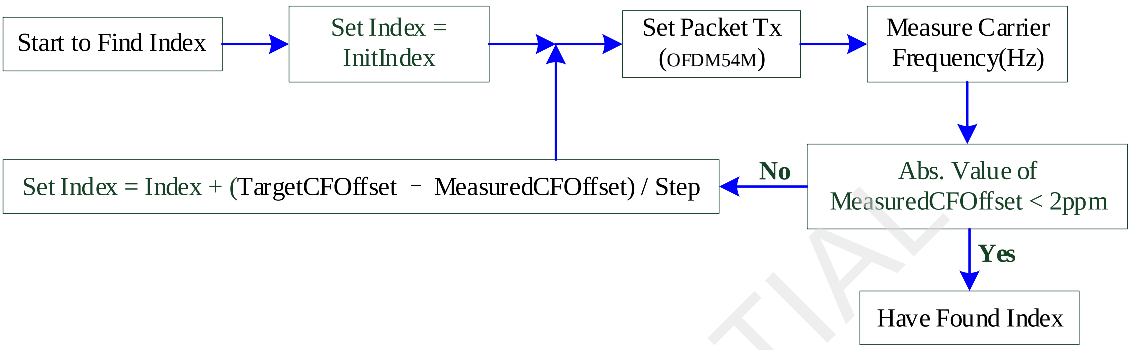


Figure 4: Finding Crystal Cap. Index Flow

Measurement environment: room temperature 25 degree
InitIndex: the default value is 0x3F. Index range is 0x0 to 0x7F.
MeasuredCFOffset: Carrier frequency measured by instrument - ideal carrier frequency target range Abs. Value of 2ppm in 2.4GHz band is about 10KHz(±5KHz).
TargetCFOffset: Generally is 0~-5ppm ppm, The reason for minus ppm was the frequency deviation could be under 15ppm at high temperature 80 degree environment.
Step: The value will be different from modules dominated by external capacitor beside the crystal, so it must be able to modify simply in the initial file of test program. By experience, the value variation is about +1 ~ +1.5KHz. The value should be checked by hardware RD and be filled in the setup file of test program. The plus symbol means that the crystal Cap. index and carrier frequency is positive-dependent (The larger index is relative to major carrier frequency).
The command is shown as below:
Step 1:

```
iwpriv mp_channel 7
iwpriv mp_rate 108                // OFDM54M rate
iwpriv mp_txpower dbm=18
iwpriv mp_phypara xcap=64
iwpriv mp_hwtx period=100,len=1500,count=0
```

Step 2:
To measure frequency error (ppm), Freq_Err.
Freq_Err should be limited to ±2ppm.
If Freq_Err is between ±2ppm, it means the Index_cry is correct.
If the Freq_Err is out of ±2ppm, try the algorithm below to find next index_cry_next until Freq_Err is between ±2ppm.

$$\text{index\_cry\_next} = \text{Index\_cry} - \frac{Freq_{Err} \times 2442}{2500}$$

Example:
Set Index_cry =63, measured Freq_Err = -23.49ppm. How to find next index?
The next index is

$$\text{index\_cry\_next} = 63 - \frac{(-23.49) \times 2442}{2500} \approx 86$$

Step 3:
The index_cry_next must be rounded to an integer, and use the following command to update index:

```
iwpriv mp_phypara xcap=index_cry_next
```

Example:

```
iwpriv mp_phypara xcap=86   //update crystal calibration value to MP
```

Repeat Step2, Step3 until the measured Freq_Err is limited to ±2ppm and note down Index_cry_ok. And Index_cry_ok should be updated to OTP after calibration. The range of CrystalCap is 0x0~0x7F.

Stop Tx command is as below:

```
iwpriv mp_hwtx stop
```

3.1.3 TSSI calibration and thermal

The following flow will show the TSSI calibration flow, criterion, and sample command.

3.1.3.1 TSSI DE locations in OTP

There are the TSSI DE for each channel group with PHY data rate.
The TSSI DE value and thermal value should be calibrated and updated to OTP.

2.4Ghz TSSI DE Location						
	Group 1 CH1 – CH2	**Group 2 CH3 – CH5**	**Group 3 CH6 – CH8**	**Group 4 CH9 – CH11**	**Group 5 CH12 – CH13**	**Group 6 CH14**
MCS7	0x26[7:0]	0x27[7:0]	0x28[7:0]	0x29[7:0]	0x2A[7:0]	
CCK	0x20[7:0]	0x21[7:0]	0x22[7:0]	0x23[7:0]	0x24[7:0]	0x25[7:0]
5Ghz TSSI DE Location						
	Group 7 CH36 – CH40	**Group 8 CH44 – CH48**	**Group 9 CH52 – CH56**	**Group 10 CH60 – CH64**		
MCS7	**0x32[7:0]**	**0x33[7:0]**	**0x34[7:0]**	**0x35[7:0]**		
	Group 11 CH100 – CH104	Group 12 CH108 – CH112	Group 13 CH116 – CH120	Group 14 CH124 – CH128	Group 15 CH132 – CH136	Group 16 CH140 – CH144
MCS7	**0x36[7:0]**	**0x37[7:0]**	**0x38[7:0]**	**0x39[7:0]**	**0x3A[7:0]**	**0x3B[7:0]**
	Group 17 CH149 – CH153	Group 18 CH157 – CH161	Group 19 CH165 – CH169	Group 20 CH173 – CH177		
MCS7	0x3C[7:0]	0x3D[7:0]	0x3E[7:0]	0x3F[7:0]		

Thermal meter	**0xCA[7:0]**

Table 2: TSSI DE and Thermal meter offset in OTP

3.1.3.2 Defined target power

According to EMI/EMC regulatory IEEE TX EVM / Spectrum Mask requirement, the target power of each channel group and PHY data rate can be defined.
Assuming that all the channels have the same target power for each PHY data rate, the recommended target power is listed as below :

	Data Rate	**HE-MCS9**	**VHT-MCS8**	**HT-MCS7**	**54M**	**CCK**
2.4G	Target Power	15dBm	16dBm	17dBm	18dBm	20dBm
5G	Target Power	14dBm	15dBm	16dBm	17dBm	

Table 3: The recommended target power

The target power in the table is RTK default value. The value can be defined by customer.
Power-by-rate table and target power must be the same value.

NOTE

If there is any concern, or want to know the more details, please contact Realtek FAE.

3.1.3.3 TSSI calibration flow

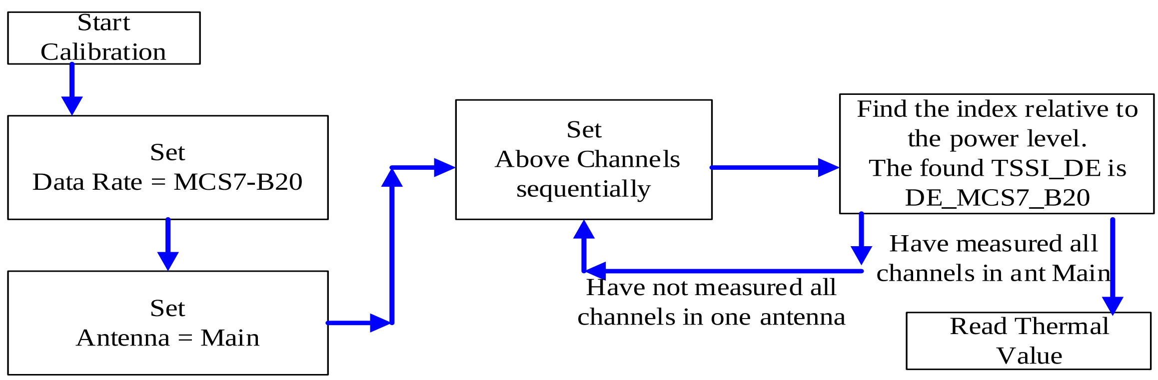


Figure 5: TSSI calibration flow

3.1.3.4 TSSI calibration command and sample

The following command take “Channel 7, HT20, HTMCS7” as example.
Step 1 : MP start and enter TSSI calibration mode.

```
iwpriv mp_rfe 7        //Set rfe type, default value is 7.
iwpriv mp_start
iwpriv mp_antsw 1      //Select 1/2 antenna(according to rfe_type),default is 1(main).
iwpriv mp_channel 7
iwpriv mp_rate 135
//rate: sets the rate of Tx packets (i.e., 2 for 1M, 4 for 2M, 11 for 5.5M, …, 108 for 54M, 128 for MCS0, 129 for MCS1, …, 135 for MCS7, 208 for HE-MCS0, 209 for HE-MCS1, …, 217 for HE-MCS9)
iwpriv mp_pwrtrk 2              //TSSI calibration mode
iwpriv mp_hwtx period=100,len=1500,count=0  //start HW Tx
```

NOTE

About HW Tx function parameters detail, please refer to appendix.

Step 3 : Measure Tx power by instrument.

```
iwpriv mp_txpower dbm=17     //set Tx target power as 17dBm
```

Step 4 : Get and record the DE value after the measured power had fit target power.

```
iwpriv mp_get_tsside adbm=17.00,pwr=15.87
           //target power = 17dbm, instrument measured power=15.87dbm
```

CMD return: patha=247 hex=0xf7.
(measured power 15.87dbm – target power 17 dbm=-1.13db, TSSI_DE = -1.13db/step 0.125db = -9(dec)=0xF7)
Record the hex:f7 DE value. Find out all the TSSI DE value for each channel group and each PHY data rate. There is a recommended interpolation method which described at section 3.1.3.5.
The DE value should be updated to HW OTP. Or the DE value can be written in driver fake OTP map first, and be updated to HW OTP after WiFi performance verification.

```
iwpriv efuse_set wlwfake,0x28,0xF7  //Set TSSI_DE(0xF7) to fake address 0x28
```

Step 5 : Before measuring the next channel, stop HW Tx.

```
iwpriv mp_hwtx stop            //Stop HW packet Tx
```

3.1.3.5 Interpolation for finding TSSI DE

Theoretically, we need to measure all value defined in OTP to calibrate the Tx power level. But since it costs too much time, we only measure several channels with MCS7-B20 signal and figure out the other non-measured value by interpolation method.
Usually, the recommended measured channels are list below:

2G Band	5G Band1	5G Band2	5G Band3	5G Band4

CH4	CH10	CH36	CH44	CH52	CH60	CH100	CH124	CH140	CH149	CH173

Table 4: The recommended measured channel for Tx power calibration

Example:
If the measured DE in CH4 is -1 and the measured DE in CH10 is 1, all 2G group MCS7-B20 TSSI DE is shown as below:
CH4 is group 2, CH10 is group 4.

Group 1	**Group 2**	Group 3	**Group 4**	Group 5
-2 (Calculated by Interpolation)	**-1 (Measured)**	0 (Calculated by Interpolation)	**1 (Measured)**	2 (Calculated by Interpolation)

Table 5: The example of finding TSSI DE in 2G band by interpolation

NOTE

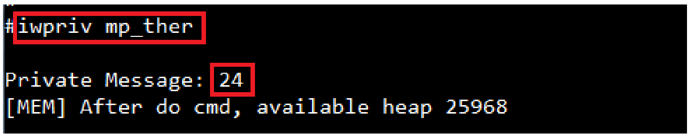
The calibrated HEX value was 2'complement! (0xFF = -1)

We have measured all groups of 5G band 1 and 5G band 2, so just use the interpolation for 5G band 3 and 5G band4.

3.1.3.6 Read Thermal Meter

Normal driver will load thermal meter to do power tracking. So this value must be filled on correct calibration data location. Use MP command as below to get thermal meter value:

```
iwpriv mp_ther
```



The timing to record Thermal value is in the last channel calibration:

Figure 6: record Thermal value flow

3.1.3.7 Write and read fake OTP

The DE value and thermal value can be written in driver fake OTP map first, and be updated to HW OTP after verification.
The following MP command only send one byte data to the fake OTP:

```
iwpriv efuse_set wlwfake,ADDR,VALUE
                //Set value to fake OTP offset(ADDR range 0x20~0x2A,0x32~0x3F,0xCA)
```

The following MP command can send 11 bytes data(for 2G) or 14 bytes data(for 5G)to the fake OTP:

```
iwpriv efuse_set wlwfake_2g,DE_VALUE1,DE_VALUE2,… DE_VALUE11
                        //from 0x20 to 0x2A, total 11 bytes data
iwpriv efuse_set wlwfake_5g,DE_VALUE1,DE_VALUE2,… DE_VALUE14
                        //from 0x32 to 0x3F, total 14 bytes data
```

The following MP command can send one byte thermal value to the fake OTP:

```
iwpriv efuse_set wlwfake,CA,THER_VAL  //Set THER_VAL to fake OTP offset(CA)
```

All the TSSI DE value and thermal value(show in table2, total 26 bytes) should be written in fake OTP before WiFi Tx performance verification.

In addition, the following MP command can read data from fake OTP:

```
iwpriv efuse_get wlwfake_all   //Read all fake OTP data
iwpriv efuse_get wlwfake_single,ADDR //Read only from one fake OTP offset(ADDR)
```

3.1.4 Rx gain calibration

The following flow will show the Rx gain calibration flow and sample command.

3.1.4.1 Rx gain offset locations in OTP

There are the Rx gain offset for each channel group with PHY data rate.
The Rx gain offset should be calibrated and updated to OTP.

	2.4Ghz Rx gain location (main path)	**2.4Ghz Rx gain location(aux path)**
OFDM	0x144[7:4]	0x144[3:0]
CCK	0x146[7:4]	0x146[3:0]
	5Ghz Rx gain location (main path)	**5Ghz Rx gain location (aux path)**
Low Band CH36~CH64	0x148[7:4]	0x148[3:0]
Middle Band CH100~CH144	0x14A[7:4]	0x14A[3:0]
High Band CH149~CH177	0x14C[7:4]	0x14C[3:0]

Table 6: Rx gain offset in OTP

3.1.4.2 Rx gain calibration flow

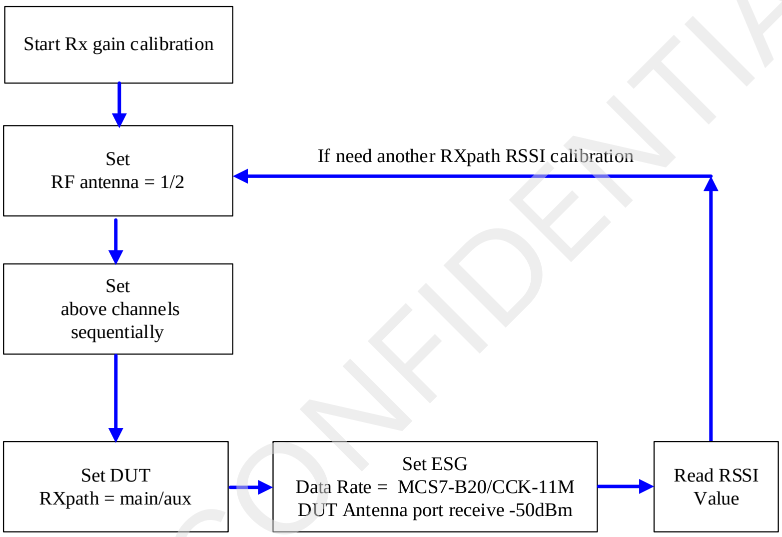


Figure 7: Rx gain calibration flow

The recommended measured channels are listed below:

2G Band		**5G Low Band**		**5G Middle Band**		**5G High Band**	
CH3	CH11	CH40	CH60	CH104	CH140	CH153	CH173

Table 7: Rx gain calibration channels

3.1.4.3 Rx gain calibration command and sample

Step 1: Set the same BW, Channel, RATE for DUT and ESG.

```
iwpriv mp_rxpath main   //Select rx path = main/aux
```

Step 2: ESG set output power and make sure the DUT antenna port received -50dBm.
Make sure the signal to DUT antenna port is -50dBm.

Step 3: ESG continue TX.

Step 4: Reset RSSI offset

```
iwpriv mp_arx gaina=0,iscck=[0/1]
                             //iscck =1 when the ESG signal is CCK
```

Step 5: Call API get RSSI value.

```
iwpriv mp_arx rssi a
```

Step 6: Measure the 2 channels each group listed above, and average the two channels' RSSI values.

Step 7: If the various of the averaged RSSI value is > ±6 dB, Please check the environment setup.
(There might be one or two opportunities to retry, as there could be random interference. However, if repeated attempts continue to fail, it's essential to check the environment.)

Step 8:
Rx gain offset = RSSI Value – (-50dBm)
Record Rx gain offset value.

Example: Measure 2G Rx gain offset
Step 1: Set the same settings to DUT and ESG.
2G Rx gain calibration channel: CH3 and CH11
Take BW20M, CH3, MCS7 for example here.
Step 2: ESG set output power and make sure the DUT antenna port received -50dBm.
Make sure the signal to DUT antenna port is -50dBm.
Step 3: ESG transmits continue TX.
Step 4: Reset RSSI offset

```
iwpriv mp_arx gaina=0,iscck=0
```

Step 5: Call API get RSSI value.

```
iwpriv mp_arx rssi a
```

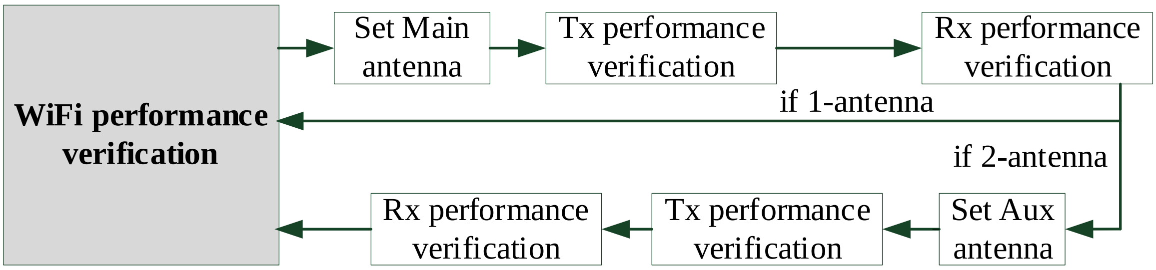
For example, MP returned CH3 RSSI = -51dBm.
Then set both ESG and DUT to channel 11, and repeat Step2~4.
MP returned CH11 RSSI = -53.5dBm
Step 6: 2.4Ghz CH3 and CH11 had measured. Averaged the two RSSI value.
CH3 RSSI = -51dBm.
CH11 RSSI = -53.5dBm
RSSI average = (-51+(-53.5))/2 = -52.25

Step 7: Check the various of the averaged RSSI value.
RSSI average = -52.25
(-52 – (-50)) < ±3 → The measuring was reliable!

Step 8: Get the Rx gain offset value.
Rx gain offset = Averaged RSSI value – (-50dBm)
(-52.25dBm) – (-50dBm) = -2.25
Rx gain offset = -2 (To ensure accurate calibration, rounding is used.)

If need another path RSSI calibration, select another rxpath, and repeat step 1 to step 8.

3.2 WiFi performance verification



3.2.1 Tx performance verification

The following flow will show the Tx performance verification flow, criterion, and sample command.

3.2.1.1 Tx performance verification criterion

Use the calibrated TSSI DE in previous step and measure Tx power, EVM, frequency offset and LO leakage to check Tx performance is ok or not. The recommended test items are listed below:

Data Rate	Antenna	Channel	Item	Criterion
HEMCS9-B20	1-Antenna DUT: Main 2-Antenna DUT: Aux	CH3	Power	Typical: 15dBm, Acceptable Range: ±1dB
			EVM	< -32dB (Full-packet compensation)
			Freq. Err.	±10ppm
			Leakage	< -32dB total
			Mask	IEEE SPEC. defined
HTMCS7-B20	Main	CH10	Power	Typical: 17dBm, Acceptable Range: ±1dB
			EVM	< -28dB
			Freq. Err.	±10ppm
			Leakage	< -21dB total
			Mask	IEEE SPEC. defined
OFDM 54M	Main	CH1	Power	Typical: 18dBm, Acceptable Range: ±1dB
			EVM	< -25dB
			Freq. Err.	±10ppm
			Leakage	< -15dBtotal
			Mask	IEEE SPEC. defined
CCK 11M	Main	CH7	Power	Typical: 20dBm, Acceptable Range: ±1dB
			EVM	< 8%
			Freq. Err.	±10ppm
			Mask	IEEE SPEC. defined

Data Rate	Antenna	Channel	Item	Criterion
HEMCS9-B20	1-Antenna DUT: Main 2-Antenna DUT: Aux	CH52 CH112 CH161	Power	Typical: 14dBm, Acceptable Range: ±1dB
			EVM	< -32dB (Full-packet compensation)
			Freq. Err.	±10ppm
			Leakage	< -32dB total
			Mask	IEEE SPEC. defined
VHTMCS8-B20	Main	CH44 CH60 CH140 CH157	Power	Typical: 15dBm, Acceptable Range: ±1dB
			EVM	< -32dB
			Freq. Err.	±10ppm
			Leakage	< -24dBtotal
			Mask	IEEE SPEC. defined
MCS7-B20		CH116	Power	Typical: 16dBm, Acceptable Range: ±1dB
			EVM	< -28dB

	Main		**Freq. Err.**	±10ppm
			Leakage	< -21dBtotal
			Mask	IEEE SPEC. defined
OFDM 54M	Main	**CH100**	**Power**	Typical: 17dBm, Acceptable Range: ±1dB
			EVM	< -25dB
			Freq. Err.	±10ppm
			Leakage	< -15dBtotal
			Mask	IEEE SPEC. defined

Table 8: The recommended test items of WiFi Tx

NOTE

The table target power is RTK default value, the user can define target power by yourself.

3.2.1.2 Verify Tx performance

The following steps are the Tx performance verify command sample:

Step 1 : Turn the MP into TSSI verification mode

```
iwpriv mp_rfe 7          //Set rfe type, default value is 7.
iwpriv mp_start
iwpriv mp_antsw 1        //Select 1/2 antenna(according to rfe_type),default is 1(main).
```

NOTE

All the TSSI DE value and thermal value should be written in fake OTP before enter TSSI verification mode.(refer to 3.1.3.7)

```
iwpriv mp_pwrtrk 1       //TSSI verification mode
```

Step 2 : Set HW Tx and TSSI DE to the case which is going to do the verification.

Example : To verify path A, CH10, HT20, MCS7, TSSI calibrated DE = -1

```
iwpriv mp_channel 10
iwpriv mp_rate 135
                        //rate: sets the rate of Tx packets (i.e., 2 for 1M, 4 for 2M,
11 for 5.5M, …, 108 for 54M, 128 for HT-MCS0, 129 for HT-MCS1, …, 135 for HT-MCS7, 160
for VHT-MCS0, …, 168 for VHT-MCS8, 208 for HE-MCS0, 209 for HE-MCS1, …, 217 for HE-MCS9)
iwpriv mp_txpower dbm=17
iwpriv mp_set_tsside patha=255     //DE = -1 (2's complement)
iwpriv mp_hwtx period=100,len=1500,count=0
```

Step 3 : Measure Tx performance and check the performance with criterion table.

CH10 HT20 MCS7 criterion:

Data Rate	Channel	Item	Criterion
MCS7-B20	**CH10**	**Power**	Typical: 17dBm, Acceptable Range: +1/-1dB
		EVM	< -28dB
		Freq. Err.	±15ppm
		Leakage	< -20dBtotal
		Mask	IEEE spec. defined

Check whether the measured result is fit the criterion or not.

Step 4 : Before change the measuring channel, stop HW Tx.

```
iwpriv mp_hwtx stop      //Stop HW packet Tx
```

NOTE

According to rfe_type, Aux-antenna Tx performance verification is needed if it exists. The recommended test items are listed at table 8.

3.2.2 Rx performance verification

The following flow will show the Rx performance verification flow, criterion, and sample command. Rx performance verification include 2 parts.

1. Rx sensitivity verification.
2. Rx gain offset verification.

3.2.2.1 Rx sensitivity verification

3.2.2.1.1 Rx sensitivity verification criterion

Measure the DUT Rx sensitivity to check Rx performance is ok or not. The recommended test items are listed below:

2.4Ghz				
Data Rate	Antenna	Channel	Item	Criterion
HEMCS9-B20	1-Antenna DUT: Main 2-Antenna DUT: Aux	CH3	Sensitivity	< -57dBm(1R)
HTMCS7-B20	Main	CH10		< -64dBm (1R)
OFDM 54M	Main	CH1		< -65dBm (1R)
CCK 11M	Main	CH7		< -76dBm (1R)

5Ghz				
Data Rate	Antenna	Channel	Item	Criterion
HEMCS9-B20	1-Antenna DUT: Main 2-Antenna DUT: Aux	CH52 CH112 CH161	Sensitivity	< -57dBm (1R)
VHTMCS8-B20	Main	CH40 CH56 CH136 CH153		< -59dBm (1R)
MCS7-B20	Main	CH116		< -64dBm (1R)
OFDM 54M	Main	CH100		< -65dBm (1R)

Table 9: The recommended test items of WiFi Rx

The recommended packet-error-rate is < 10%(MCS, OFDM), < 8% (CCK).

NOTE

The criterion can be modified by customer requirement.

If there is any concern, or want to know the more details, please contact Realtek FAE.

3.2.2.1.2 Verify Rx sensitivity

Set the instrument to Tx packets, and DUT MP Rx the packets.
Then parsing the error rate to verify Rx performance.

The following steps are the Rx performance verify command sample :
Step 1 : Start Rx
Example : Verify path A, MCS7, bandwidth 20, CH10 Rx.

```
iwpriv mp_rfe 7          //Set rfe type, default value is 7.
iwpriv mp_start          //Start MP mode
iwpriv mp_antsw 1        //Select 1/2 TRX antenna(according to rfe_type),default is 1.
iwpriv mp_channel 10     //Set Rx channel
iwpriv mp_rxpath main    //Select rx path, main/aux
iwpriv mp_arx start      //Start Rx
iwpriv mp_reset_stats    //Reset the Rx report
```

Instrument Tx packets at HTMCS7, bandwidth 20, CH10 and adjust the Tx power let DUT input power < -61dBm (HTMCS7-B20 criteria).

Step 2 : While the instrument Tx is finished, parsing the Rx report.

```
iwpriv mp_arx phy
```

The command will report Rx OK/CRC_error counter.

Step 3 : Calculate packet-error-rate
Calculate packet-error-rate with step2 result.
The recommended packet-error-rate is < 10%(MCS, OFDM), < 8% (CCK).

Example :
(1) Instrument Tx 1000 packets at MCS7, bandwidth 20, CH10.
(2) Reports DUT had Rx 920 packets which Tx from instrument and its input power < -61dBm successfully.
(3) (1000 – 920) / 1000 = 8%
(4) 8% < 10% → Path A, MCS7, bandwidth 20, CH10 Rx verification pass!

NOTE

According to rfe_type, Aux-antenna Rx performance verification is needed if it exists. The recommended test items are listed at table 9.

3.2.2.1.3 Rx gain offset verification criterion

Measure the DUT RSSI to check whether the Rx gain offset is suitable or not.
The recommended test items are listed below:

2G Band	5G Low Band	5G Middle Band	5G High Band
CH7	CH48	CH120	CH165

Pick up a channel at each band to do Rx gain offset verification.
Rx gain verification criterion:

Data Rate	Channel	Item	Criterion
MCS7-B20	2G : CH7	RSSI	-50 ±1.5 dBm
	5GLB : CH48		
	5GMB : CH120		
	5GHB : CH165		

Table 10: Rx gain verification criterion

3.2.2.1.4 Verify Rx gain offset

Step 1: Set the same BW, Channel, RX path, RATE for DUT and ESG.

Step 2: ESG set output power and make sure the DUT antenna port received -50dBm.
Make sure the signal to DUT antenna port is -50dBm.

Step 3: ESG continue Tx.

Step 4: Set calibrated Rx gain offset value

```
iwpriv mp_arx gaina =gain offset,iscck=0/1
                                         //iscck =1 when the ESG signal is CCK
```

Step 5: Call API get RSSI value.

```
iwpriv mp_arx rssi a
```

Step 6: As the criterion, the returned RSSI value should be limited in -50 ±1.5dBm.

Step 7: The verified pass Rx gain offset value should check in to OTP.

4 WiFi/BT Interface Switch

The ATM2 series command can switch WiFi or BT MP test mode. Customers should follow this list by steps to do WiFi & BT Mass Production.

<table>
<tr><th>No</th><th>Steps</th><th>Commands/Descriptions</th><th>Band rate</th><th>Remark</th></tr>
<tr><td>1</td><td>MP start</td><td>iwpriv mp_rfe 7
iwpriv mp_start</td><td>1500000</td><td></td></tr>
<tr><td>2</td><td>WiFi Calibration & Verification</td><td>iwpriv commands</td><td>1500000</td><td>Reference doc:
AN0004 Realtek low power wi-fi mp user guide.</td></tr>
<tr><td>3</td><td>Switch to Bluetooth</td><td>ATM2=ant,s0/ATM2=ant,s1
ATM2=bt_power,on
ATM2=gnt_bt,bt
ATM2=bridge</td><td>1500000</td><td>Please refer to 5.1.1.</td></tr>
<tr><td>4</td><td>Open BT Uart interface</td><td rowspan="5">Refer to the sample codes in Realtek Bluetooth MP Kit Setup Package</td><td rowspan="5">115200</td><td rowspan="5">Reference doc: MP tool User guide for Realtek.
Sample code path:
RealeasePackage\NewBTMP\Realtek Bluetooth MP Kit Setup Package\ RTLBTAPP\SampleCode\ MPRtlBluetoothSample

Note : keep the tx gain k value and thermal value</td></tr>
<tr><td>5</td><td>BT Tx calibration (gain k, flatness k)</td></tr>
<tr><td>6</td><td>Read BT thermal meter value</td></tr>
<tr><td>7</td><td>BT TRx verification</td></tr>
<tr><td>8</td><td>Close BT Uart interface</td></tr>
<tr><td rowspan="2">9</td><td rowspan="2">Switch to Wifi</td><td>ATM2=bridge,close</td><td>115200</td><td rowspan="2">Please refer to 5.1.1.</td></tr>
<tr><td>ATM2=gnt_bt,wifi</td><td>1500000</td></tr>
<tr><td>10</td><td>Write OTP</td><td>iwpriv commands</td><td>1500000</td><td>Reference doc: Realtek RTL8730E calibration data spec.
* Write WiFi calibration data+MAC address
* Write BT calibration data+MAC address.</td></tr>
<tr><td>11</td><td>MP stop</td><td>iwpriv mp_stop</td><td>1500000</td><td></td></tr>
</table>

Table 11: WiFi & BT Mass Production flow

5 BT Mass Production Flow

The below diagram shows a global view of Bluetooth mass production flow:

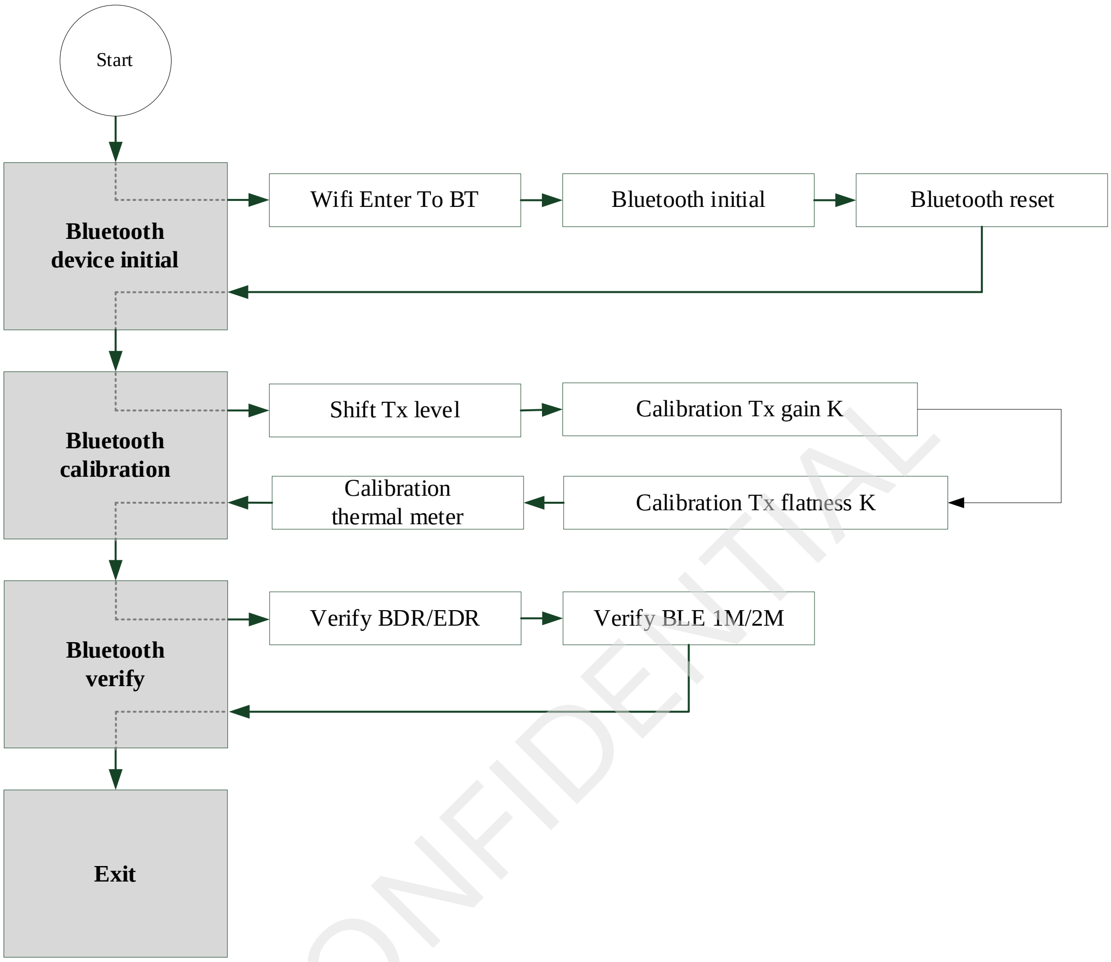
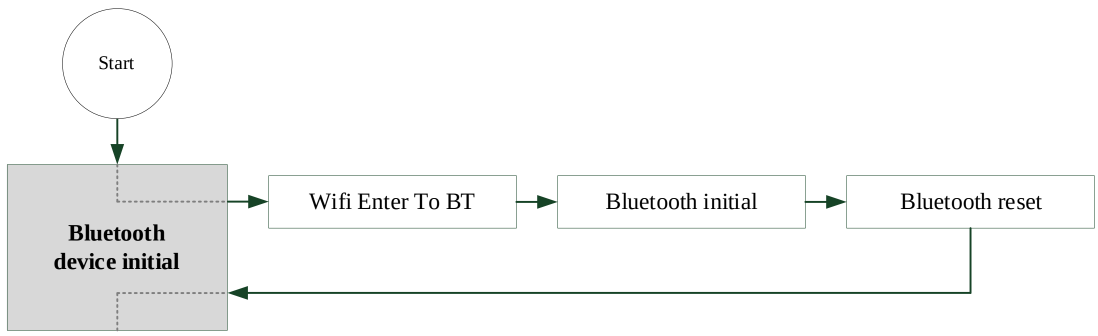


Figure 8: BT Mass Production Test Flow

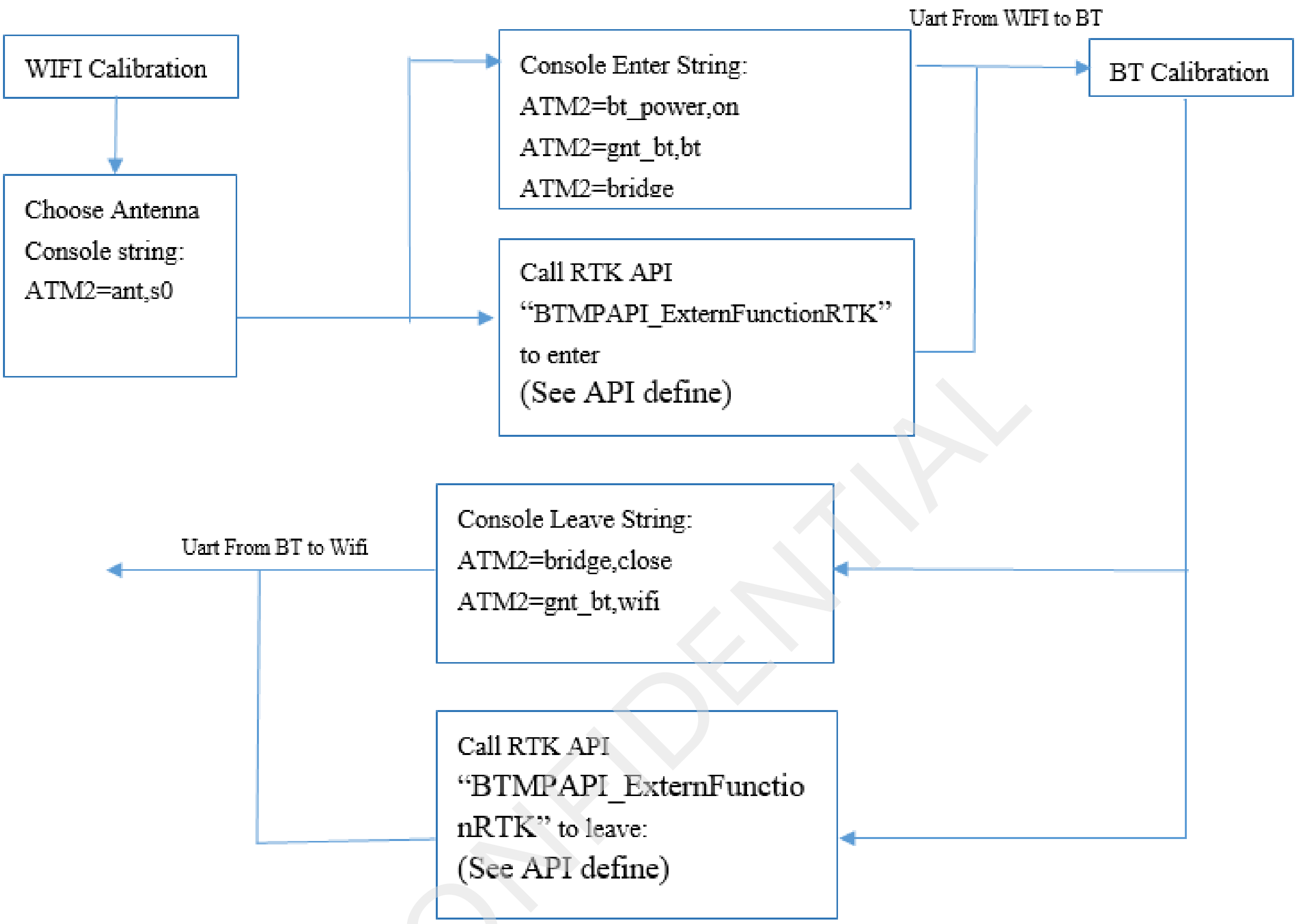
5.1 Bluetooth Device Initial

First, need to switch interface path to BT . Please go through the WIFI MP driver and tool to execute the Job. Next to competed the BT initial stage, the Bluetooth device must set to factory default.



5.1.1 Wifi Mode Enter To BT HCI Mode

There are two methods enter to bt hci mode from WLAN mode.
1. Using Console enter String.(suggestion)
2. Using RTK BT API – “BTMPAPI_ExternFunctionRTK”.



5.1.2 Bluetooth Initial: Disable TX power tracking

If platform is Windows, please use BT DLL of Realtek to set it. Use the function to enable or disable thermal power tracking for chip. A simple example to use "API_ThermalPowerTracking" of RF dynamic link library is as follows:

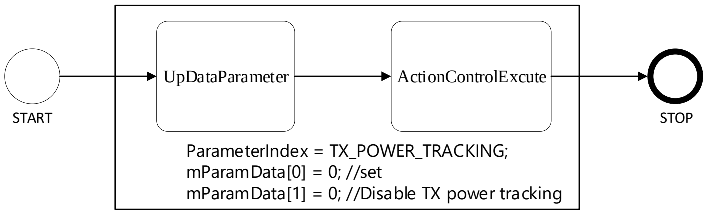


Figure 9: Disable TX power tracking flow

Sample:

```
pParam->ParameterIndex = TX_POWER_TRACKING;
pParam->mParamData[0] = 0; //set
pParam->mParamData[1] = OnOff; //on(1) or off(0)
pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
pBluetoothModule->ActionControlExcute(pBluetoothModule);
```

You can refer to "BTDevice_SetThermalPowerTracking(unsigned char OnOff) " function of sample program.

5.1.3 Bluetooth Initial: Set TX Power Level To RAM

For verify TX performance need to setting temporarily set the target TX power level. You can use the APIs below to set the device to the ram. The Tx power level please refer "Tx power Target".

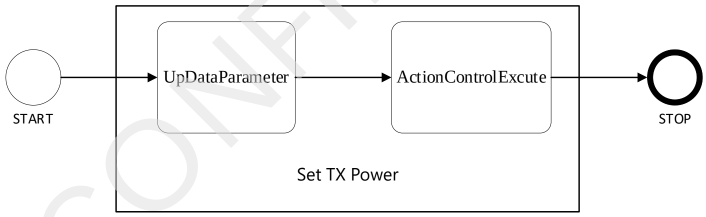


Figure 10: Set TX Power Level To RAM flow

Sample:

```
#define HCI_VENDOR_SETTING_TX_LEVEL_TO_RAM 0xFCE7

unsigned char powerlevel[512]={0};
    unsigned char pEvtBuf[512] = {0};

    powerlevel[0] = TX_level_1M;
    powerlevel[1] = TX_level_2M;
    powerlevel[2] = TX_level_3M;
    powerlevel[3] = TX_level_BLE1M;
    powerlevel[4] = TX_level_BLE2M;

    pBluetoothModule->SendHciCommandWithEvent
(pBluetoothModule,HCI_VENDOR_SETTING_TX_LEVEL_TO_RAM , LEN_5_BYTE, powerlevel, 0x0E,
pEvtBuf);
```

The tx level value, you can refer Tx power Level table. The table by different Chip.

5.1.4 Bluetooth Reset: Set TX gain cal (K) value

If you need to calibrate to targeted power, then must reset the TX gain Cal Value. If platform is Windows, please set it through BT DLL of Realtek.

The TXGainK Value is a 2's complement.

Parameter Index	Param Data length	mParamData	
		0	
TX_POWER_GAIN_K	2	0x01 (Set)	TXgain cal Value

A simple example to use "API_ThermalPowerTracking" of RF dynamic link library is as follows:

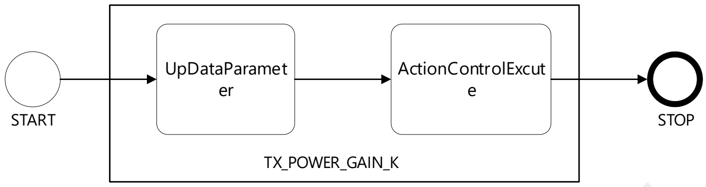


Figure 11: Set TX gain cal (K) value

Sample:

```
pParam->ParameterIndex = TX_POWER_GAIN_K;
pParam->mParamData[0] = 0x01; //set
pParam->mParamData[1] = TXgain cal Value; //set value
pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
pBluetoothModule->ActionControlExcute(pBluetoothModule);
```

If need to reset, then TXgain cal Value is 0.

5.1.5 Bluetooth Reset: Set TX flatness Value

If you need to calibrate TX power for each frequency group, then must set flatness value to zero first. If platform is Windows, please set it through BT DLL of Realtek.

The TX flatness Value is a 2's complement in each bytes.

Mode	Index	length	mParamData				
			0	1	2	3	4
4 Byte	TX_POWER_GAIN_K	5	0x02 (Set)	Flatness Value LSB		Flatness Value MSB	

Table 12: Set flatness Value command table

The flatness settings of APIs are defined below:
A simple example to use "API_ThermalPowerTracking" of RF dynamic link library is as follows:

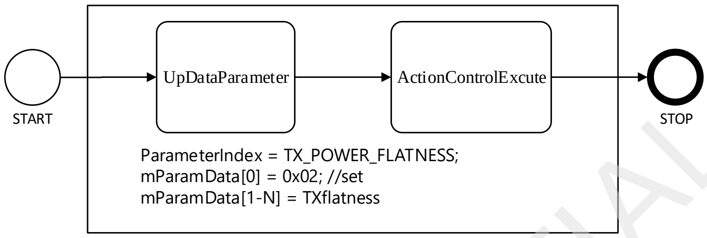


Figure 12: Set flatness Value flow

Sample:

```
pParam->ParameterIndex = TX_POWER_FLATNESS;
pParam->mParamData[0] = 0x02; //set
pParam->mParamData[1] = TXflatness &0xFF ;//LSB;
pParam->mParamData[2] = (TXflatness >>8) &0xFF; //LSB_MSB;
pParam->mParamData[3] = (TXflatness>>16) &0xFF ;//MSB_LSB;
pParam->mParamData[4] = (TXflatness >>24) &0xFF; //MSB;
pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
pBluetoothModule->ActionControlExcute(pBluetoothModule);
```

If need to reset, then the Value is 0x00000000.

5.1.6 Bluetooth Reset::Set TX path loss module Value

If you need to calibrate TX power to target Tx power, then must set the Tx path loss module value. Often in mass production, the RF of the board end is equal to the RF end of the chip, and this can be set to TX path loss module Value to 0x00.

The TX path loss module value is a 2's complement.

Parameter Index	Param Data length	mParamData	
		0(sub index)	
TX_POWER_GAIN_K	2	0x03 (Set)	TX path loss module Value

Table 13: Set TX path loss module Value Table

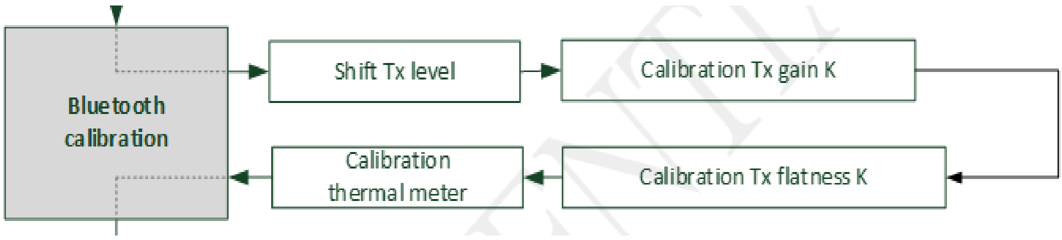
Sample:

```
pParam->ParameterIndex = TX_PATH_LOSS_MODULE;
pParam->mParamData[0] = 0x03; // sub Index = set TX_path_loss_module
pParam->mParamData[1] = TX_path_loss_module Value; //set value
pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
pBluetoothModule->ActionControlExcute(pBluetoothModule);
```

5.2 Bluetooth calibration Frequency

First, take a look at OTP content about setting of Crystal Cap. Normal driver will load this value in initial step. So this value must be well-calibrated and filled on correct OTP location.
For this Chip, Bluetooth follows set of WLAN, so only need to correct WLAN frequency offset.

5.3 Bluetooth calibration Tx Power



Calibration Tx Power environment architecture for this process is defined.

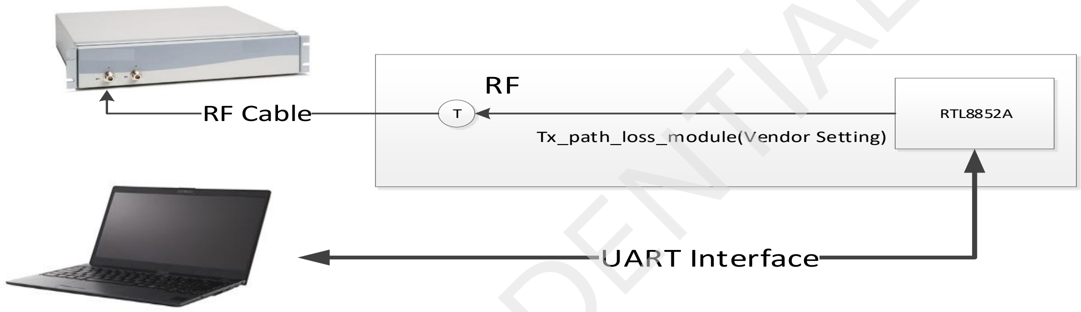
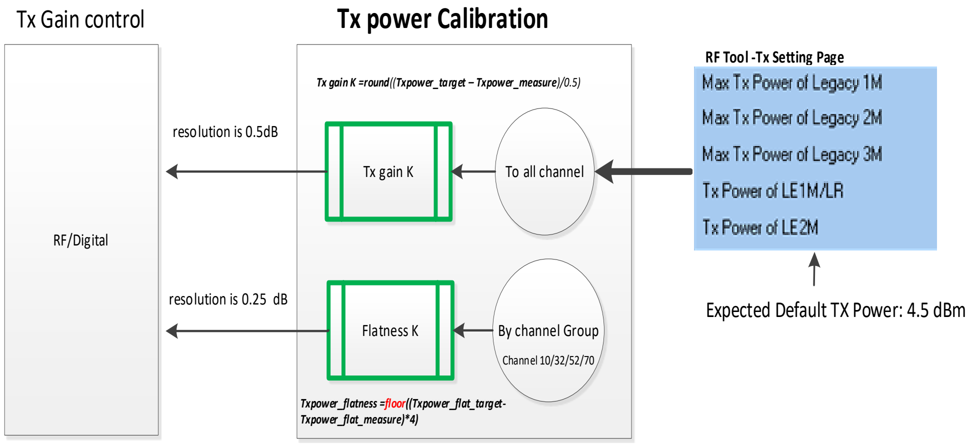


Figure 13: Bluetooth calibration Tx Power schematic

This simple diagrammatic sketch is as follows:



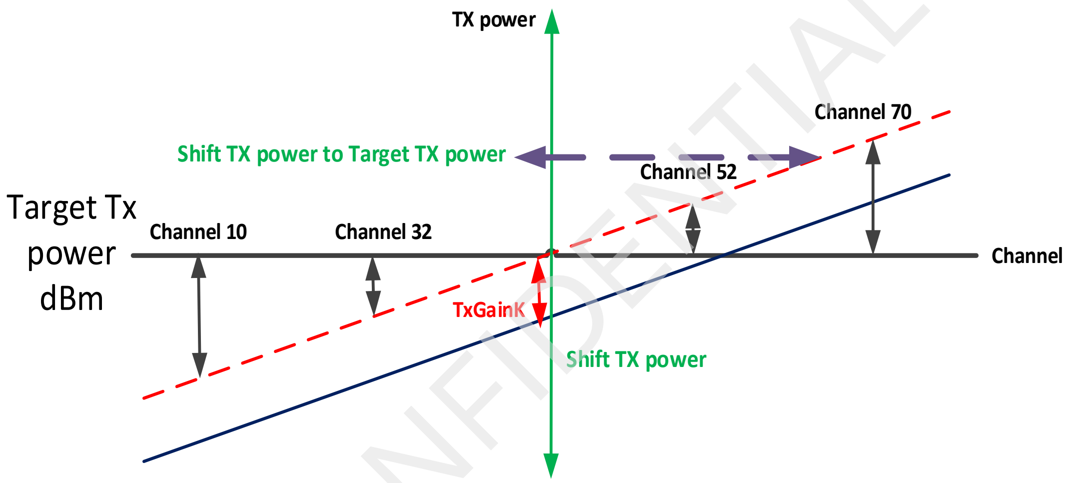


Figure 14: Bluetooth TX Power Calibration Diagram

5.3.1 Disable TX power tracking

If platform is Windows, please use BT DLL of Realtek to set it. Use the function to enable or disable thermal power tracking for chip. A simple example to use "API_ThermalPowerTracking" of RF dynamic link library is as follows:

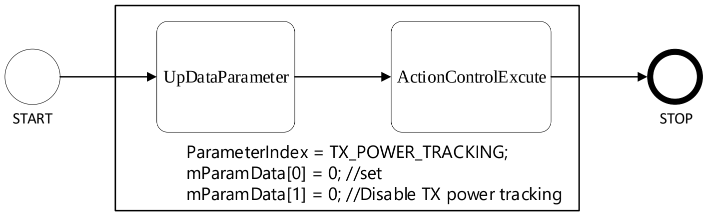


Figure 15: Disable TX power tracking flow

```
pParam->ParameterIndex = TX_POWER_TRACKING;
pParam->mParamData[0] = 0; //set
pParam->mParamData[1] = OnOff; //on(1) or off(0)
pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
if (pBluetoothModule->ActionControlExcute(pBluetoothModule) !=0)
{
    goto error;
}
```

You can refer to "BTDevice_SetThermalPowerTracking(unsigned char OnOff) " function of sample program.

5.3.2 Set Power TX Level to RAM

First, to shift the Tx power level to target power. The flow show as:

Figure 16: Set Power Tx Level flow

When the calibration is completed, please write the set target tx power value to the device. The Tx power level please refer Tx power Setting table.

5.3.3 Calibration TX gain K

Here demonstrate the flow about calibrating TX power to target power. Following plot adapted power target TX power as an example but we suggest using desired target power as calibrated power in this step.

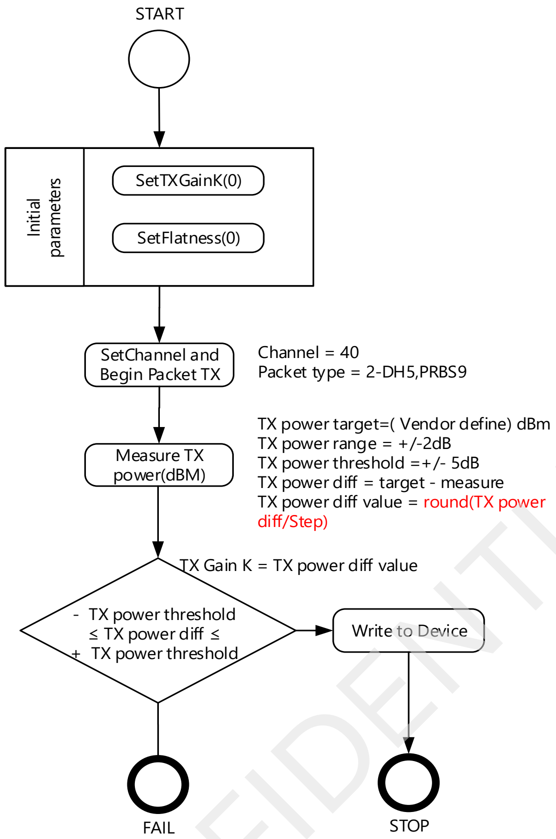


Figure 17: Calibration to Target Power

5.3.4 Calibration TX flatness K

Following plot adapted target tx power as an example but we also suggest using desired target power as calibrated power in this step.

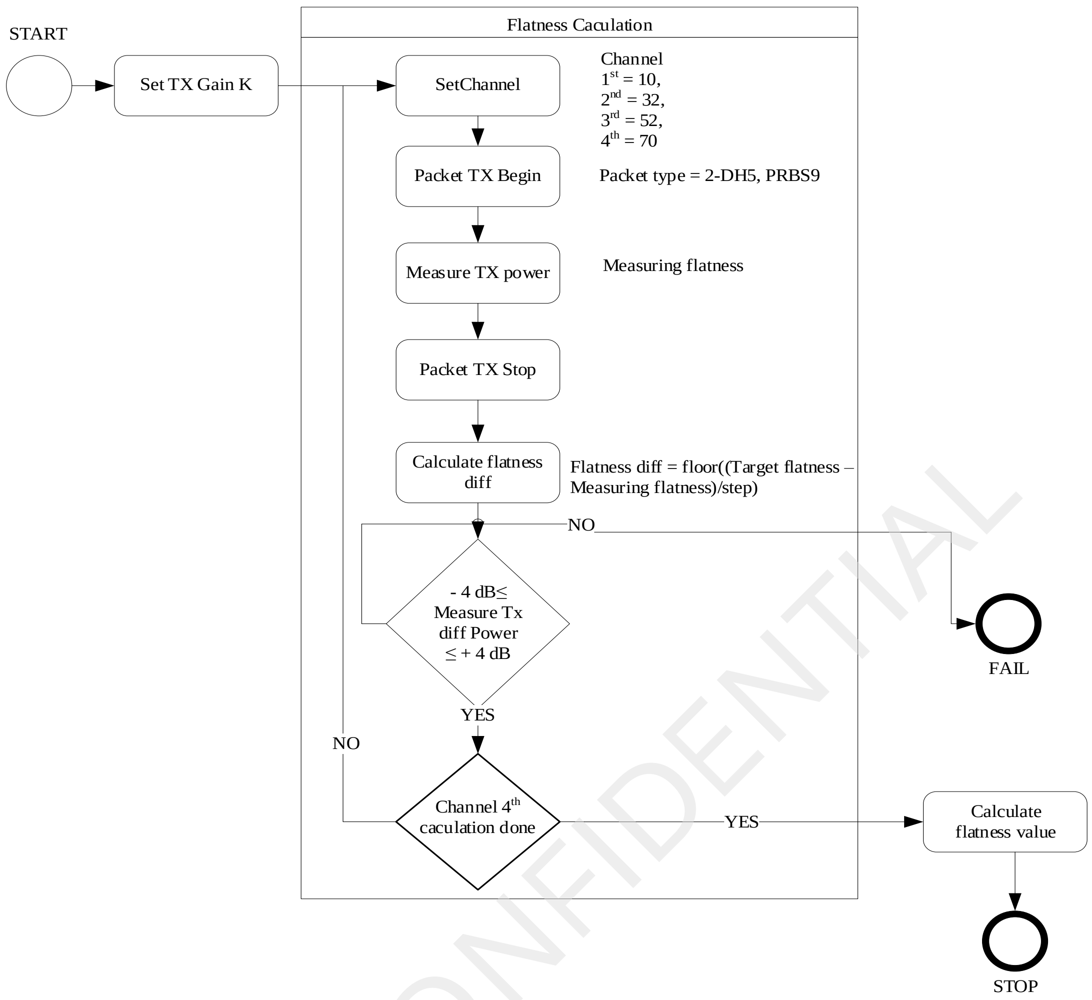


Figure 18: Bluetooth TX Power Flatness Calibration flow

The description of the process parameters:
TXpower_flat_measure[0~3]: Measure TX power.
TXpower_flat_target: Target flatness power which is set to target tx power.
TXpower_flat_diff[0~3]: Measure each channel's difference between TX power measurement and target flatness TXpower. Defined as :
TXpower_flat_diff[0~3] =floor((TXpower_flat_target flatness –measure flatness)/step)
Calculate Tx power flatness value:
If flatness is 4 byte mode, then step is 0.25 dB.

For example:
The flatness is 4 byte mode, then step is 0.25 dBm.
The TXpower_flatness value is defined as:

Bits	**0~7**	**8~15**	**16~23**	**24~31**
define	Low Channel Flatness, Ch2 ~ 21	Mid Low Channel Flatness Ch22~ch43	Mid High Channel, Flatness ch44 ~ 61	High Channel Flatness ch62 ~ 76

The TX power flatness value define table(4byte)

Write to OTP the step is 0.0625 dB, so
Flatness value (to OTP) = Flatness value(to ram) *4.

For example:

Channel	**Measured Tx power**	**Tx power flat_diff Calculate**	**Tx power flat_diff**	**Value**	**Flatness Value (Set to RAM)**	**Flatness Value *4 (Write to OTP)**
10	-0.6	floor((0-(-0.6))/0.25)	2	0x02 -> 0x08	0xfbfd0102	0xecf40408
32	-0.3	floor((0-(-0.3))/0.25)	1	0x01-> 0x04		
52	0.7	floor((0-(0.7))/0.25)	-3	0Xfd-> 0Xf4		
70	1.2	floor((0-(1.2))/0.25)	-5	0xfb-> 0xec		

Table 14: The TX power flatness value define sample table (4byte)

5.3.5 Calibration Thermal (Tmeter) vaule

Normal driver will load thermal meter to do power tracking. So this value must be filled on correct OTP location. Use the command below to read Bluetooth thermal meter once.

If platform is Windows, please through BT DLL of Realtek to obtain raw thermal value. Please refer below sample code of use DLL:

Sample:

```
BT_DEVICE_REPORT    *pModuleBtReport=&mBT_DEVICE_REPORT;
//Read Tmeter
pBluetoothModule->ActionReport(pBluetoothModule,REPORT_THERMAL,pModuleBtReport);
```

NOTE

The Thermal Value is RAW Data of Realtek define.

5.4 Bluetooth verify flow

To measure the DUT Bluetooth legacy TX power/initial Carrier offset/modulation characteristics to check Tx performance is ok or not. The DUT parameter setting is determined by the pattern of the instrument.

5.4.1 Verify Bluetooth Legacy Tx Performance

Below SPEC is defined by using 7 dBm(BR/EDR) and 7dBm(BLE) as target power at normal temperature 25 degree C. For vender's Tx verification, please adapt target power value as your own SPEC. Bluetooth legacy TX criterion is shown as below:

	Test Item	Test Item	Packet Type	Channel	Criterion
Verify Tx DH1					Bluetooth Spec. S0:Target tx power = 7dBm S1:Target tx power = 8dBm
	Maximum Output Power	Average Power	DH1	CH2-2404MHZ	7 dBm NOTE: variation =target power value +/- 2 dB*
				CH40-2442MHZ	
				CH76-2478MHZ	
	Modulation Characteristics	Delta F1 Avg.	DH1	CH2-2404MHZ	140KHz ~ 175KHz
		Delta F2 Max.		CH40-2442MHZ	> 115KHz
		Modulation Index		CH76-2478MHZ	> 0.8
	Initial Carrier Frequency Error		DH1	CH40-2442MHZ	-20KHz ~ 20KHz
Verify Tx 2DH5/ 3DH1	Maximum Output Power	Average Power	3DH1	CH2-2404MHZ	7dBm NOTE: variation =target power value +/- 2 dB*
				CH40-2442MHZ	
				CH76-2478MHZ	
	Modulation Characteristics	RMS DEVM	2DH5/ 3DH1	CH2-2404MHZ	0.13
		Peak DEVM		CH40-2442MHZ	0.25
		99% DEVM		CH76-2478MHZ	0.20
	Initial Carrier Frequency Error		3DH1	CH2-2404MHZ	-20KHz ~ 20KHz
				CH40-2442MHZ	
				CH76-2478MHZ	

Table 15: Bluetooth legacy TX criterion

NOTE

If Tx power EIRP> 10 dBm, you need to enable the function of Adaptivity.

The variation = target power variation +/- 2 dB. (Only available after doing Tx calibration flow)

For Verify Bluetooth performance, there is a simple example of parameter setting below:

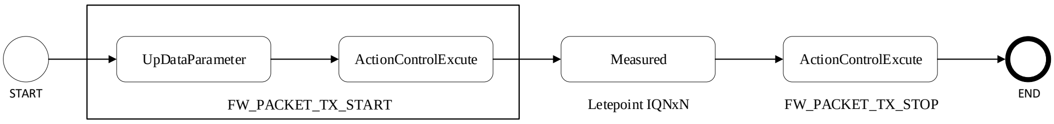


Figure 19 Verify Bluetooth Legacy Tx Performance flow

Step by step command of use API's:
(1) Enter MP Mode and download patch code
(2) Set Parameter:

Packtype	Test Item	Parameters	

DH1	Maximum Power	//channel : 0x02;//0x28 ,0x4c
		mChannelNumber=0x02;
		mPacketType=0x00;
		mPayloadType=0x07;
		mPacketHeader=0x33820;
			mTxPacketCount=0x00;
			mWhiteningCoeffValue=0x80;
			mTxGainIndex=0x00;
			mHitTarget=0x0000009e8b33

DH1	Delta F1	//channel : 0x02;//0x28 ,0x4c
		mChannelNumber=0x02;
		mPacketType=0x00;
		mPayloadType=0x05;
		mPacketHeader=0x33820;	

DH1	Delta F2	//channel : 0x02;//0x28 ,0x4c
		mChannelNumber=0x02;
		mPacketType=0x00;
		mPayloadType=0x02;
		mPacketHeader=0x33820;	

2DH5	ALL	//channel : 0x02;//0x28 ,0x4c
		mChannelNumber=0x02;
		mPacketType=0x05;
		mPayloadType=0x07;
		mPacketHeader=0x3F870;	

3DH1	ALL	//channel : 0x02;//0x28 ,0x4c
		mChannelNumber=0x02;
		mPacketType=0x06;
		mPayloadType=0x07;
		mPacketHeader=0x15C40;	

(3) Run Packet Tx
(4) Measured by Bluetooth test instrument (ex. Letepoint IQNxN)
PS. "ActionReport" should be called every 200ms.
(5) Stop Packet Tx
PS. If you need to test other parameters, please stop packet tx and go back to step 2

5.4.2 Verify Bluetooth BLE4.0 Tx Performance

To measure the DUT BLE TX power and modulation index to check BLE TX performance is ok or not.

Bluetooth BLE TX criterion is shown as below:

	Test Item	Sub Test Item	Payload Type	Channel	Criterion
Verify BLE Tx					Bluetooth Spec. S0:Target tx power = 7dBm S1:Target tx power = 8dBm
	BLE Output Power	Average Power	PRBS9	CH1-2404MHZ CH20-2442MHZ	7dBm NOTE: variation = target power value +/- 2 dB*

				CH38-2478MHZ	
	Modulation Characteristics	Delta F1 Avg.	BT_PAYLOAD_TYPE_1111_0000	CH20-2442MHZ	225 kHz ~ 275 kHz
		Delta F2 Max.	BT_PAYLOAD_TYPE_1010		≥ 185 kHz
		Modulation Index	None		≥ 0.8

Table 16: Bluetooth BLE TX criterion(BT4.0)

NOTE

variation = target power variation +/- 2 dB. (only available after doing Tx calibration flow)

For example, the step by step:

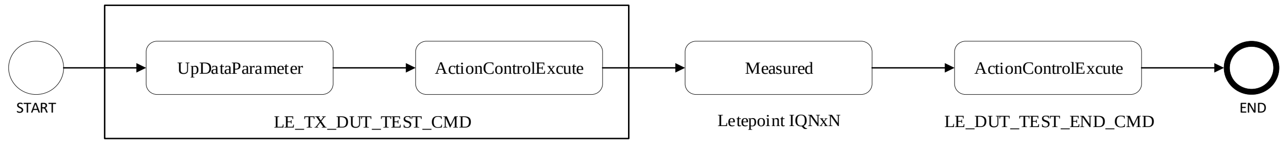


Figure 20 Verify Bluetooth LE TX Performance flow(BT4.0)

If interface of device is UART, then the step by step commands:

(1) Enter MP Mode and download patch code

(2) Set Parameter :

Packtype	Test Item	Param

BLE	Average Power	mChannelNumber=0x01;
		//(change channel :0x14,0x26)
		
		mPayloadType=0x00;
		//Payload type = PRBS9
		
		mTxGainIndex=0x00;

BLE	Delta F1	mChannelNumber=0x01;
		//(change channel :0x14,0x26)
		
		mPayloadType=0x01;
		//Payload type = 11110000
		
		mTxGainIndex=0x00;

BLE	Delta F2	mChannelNumber=0x01;
		//(change channel :0x14,0x26)
		
		mPayloadType=0x02; //Payload type =10101010
		
		mTxGainIndex=0x00;

5.4.3 Verify Bluetooth BLE5.0 Tx Performance

To measure the DUT BLE 5.0 TX power and modulation index to check BLE TX performance is ok or not. Bluetooth BLE 5.0 TX criterion is shown as below. For vender's TX verification, please adapt target power value as your own spec. The Target TX power spec depends on different chips. BLE 5.0 TX criterion is shown as below:

PHY Spec	Test Item	Sub Test Item	Payload Type	Channel	Criterion
					Bluetooth Spec. S0:Target tx power = 7dBm S1:Target tx power = 7dBm
2M	BLE Output Power	Average Power	PRBS9	CH1-2404MHZ	7 dBm NOTE: Max variation = target power value +/- 2 dB*
LR S2					
LR S8					
2M	Carrier freq. offset & drift	None	PRBS9	CH20-2442MHZ	2M: < 20 kHz
LR S2			PRBS9		S8: < 19.2 kHz

LR S8			PRBS9		
2M	Modulation Characteristics	Delta F1 Avg.	BT_PAYLOAD_TYPE_ 1111_0000	CH38-2478MHZ	2M: 450 ~ 550 kHz S8: 225 ~ 275 kHz
LR S2		Delta F2 Max.	BT_PAYLOAD_TYPE_ 1010		2M: > 370 kHz
LR S8		F2avg/F1avg	None		2M: > 0.8

Table 17: Bluetooth BLE 5.0 TX criterion

For this test must to setting current tx power by Calibrates TX power.
For example:

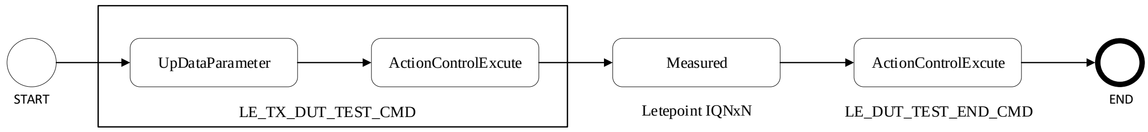


Figure 21: Verify Bluetooth LE TX Performance flow(BT5.0)

For example, the step by step :
(1) Set Parameter :

Test Item	Channel = 1/20/38

Average Power	pParam ->ParameterIndex =LE_TX_DUT_TEST_CMD
	pParam -> mChannelNumber=0x01;
	// change channel :0x14,0x26
	pParam ->mPayloadType=0x00; //Payload type = PRBS9
	pParam->PHY = 2 ; // 1:LE 1M 2:LE 2M 3: LE Coded PHY
	pParam->ModulationIndex = 1;
	// 0: standard modulation 1: stable modulation

Delta F1	pParam ->ParameterIndex =LE_TX_DUT_TEST_CMD
	pParam ->mChannelNumber=0x01 ;
	//change channel : 0x14,0x26
	pParam ->mPayloadType=0x01; //Payload type =00001111
	pParam->PHY = 2 ; // 1:LE 1M 2:LE 2M 3: LE Coded PHY
	pParam->ModulationIndex = 1;
	// 0: standard modulation 1: stable modulation

Delta F2	pParam ->ParameterIndex =LE_TX_DUT_TEST_CMD
	pParam ->mChannelNumber=0x01;
	// change channel : 0x14,0x26
	pParam ->mPayloadType=0x02; //Payload type =10101010
	pParam->PHY = 2 ; // 1:LE 1M 2:LE 2M 3: LE Coded PHY
	pParam->ModulationIndex = 1;// 0: standard modulation 1: stable modulation

pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
(2)Execute active job - Run LE DUT TX
pBluetoothModule->ActionControlExcute(pBluetoothModule);
(3) Measured by Bluetooth test instrument(ex. Letepoint IQNxN)
(4) Execute active job - Run Packet Stop
Ex.
ParameterIndex = FW_PACKET_TX_STOP
pBluetoothModule->ActionControlExcute(pBluetoothModule);

5.4.4 Verify Bluetooth Legacy Rx Performance

Measure the DUT Rx sensitivity to check Rx performance is ok or not. The Rx performance test can be measured in Signaling mode (ex: Anritsu 8852B, Agilent N4010A) or Non-Signaling mode (ex:LitePoint IQNxN). Bluetooth Rx criterion is shown as below:

Verify Bluetooth Rx	**Test Item**	**Packet Type**	**Criterion**
			Bluetooth Spec
	Sensitivity	DH1 and 3DH5	< -70dBm

Table 18: Verify Bluetooth Legacy Rx Performance Spec

For final MP, Rx can just test DH1, 3DH5 BER at sensitivity criterion power level at channel 2 and 76 to reduce time. All Bluetooth Rx criterion is shown as:

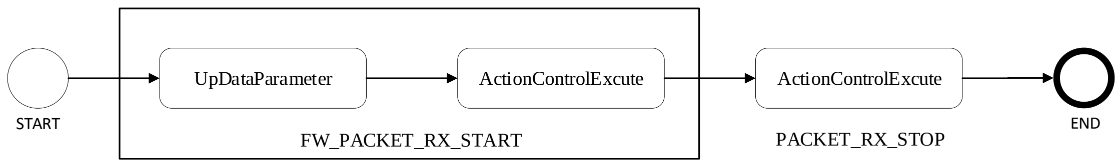


Figure 22: Verify Bluetooth Legacy Rx api's flow

Test Item		**Criterion(Bluetooth Spec)**
Channel	Packet type	< -70dBm
2	DH1	< -70dBm
40	DH1	< -70dBm
40	2DH5	< -70dBm
2	3DH1	< -70dBm
76	3DH1	< -70dBm

Figure 23: The recommended test items of Bluetooth Rx

(1) Enter MP Mode and download patch code

(2) Set Parameter : mHitTarget is BD_ADDR of Bd_addr of instrument pattern

Channel	**Packet type**	

2	DH1	mChannelNumber=0x02;
		mPacketType=0x00;
		mPayloadType=0x07;
		mWhiteningCoeffValue=0x80;
		mPacketHeader=0x33820;
		mHitTarget=0x000000c6967e

40	DH1	mChannelNumber=0x28;
		mPacketType=0x00;
		mPayloadType=0x07;
		mWhiteningCoeffValue=0x80;
		mPacketHeader=0x33820;
		mHitTarget=0x000000c6967e

40	2DH5	mChannelNumber=0x28;
		mPacketType=0x05;
		mPayloadType=0x07;
		mWhiteningCoeffValue=0x80;
		mPacketHeader=0x3F870;
		mHitTarget=0x000000c6967e

2	3DH1	mChannelNumber=0x02;
		mPacketType=0x06;
		mPayloadType=0x07;
		mWhiteningCoeffValue=0x80;
		mPacketHeader=0x15C40;
		mHitTarget=0x000000c6967e

76	3DH1	mChannelNumber=0x4c;

		mPacketType=0x06;
		mPayloadType=0x07;
		mWhiteningCoeffValue=0x80;
		mPacketHeader=0x15C40;
		mHitTarget=0x000000c6967e

(3)To setting Parameter with the Bluetooth test instrument. Bluetooth test instrument begin transmit..
(4)Run Packet Rx
(5) Report Received Result.
PS. “ActionReport ” should be called every 200ms.
(6)Stop Packet Rx
PS. If you need to test other parameters, please stop packet Rx and go back to step 2.

5.4.5 Verify Bluetooth BLE4.0 Rx Performance

Measure the DUT BLE Rx sensitivity to check Rx performance is ok or not. The Bluetooth Rx criterion is shown as below:

Channel	Payload Type	BLE sensitivity limit	Criterion
			Bluetooth Spec
CH1-2404MHZ	PRBS9	PER<= 30.800 %	-85dBm
CH20-2442MHZ			
CH38-2478MHZ			

Table 19: The recommended test items of Bluetooth BLE Rx

To calculate the Packet Error Rate(PER)
PER%= 100*(1-(Packets Received /Packets Send))

The step by step list below:

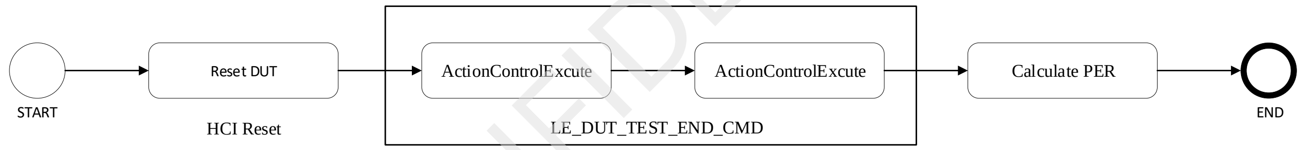


Figure 24: The recommended test items of Bluetooth BLE Rx flow

(1) Enter MP Mode and download patch code
(2) Set Parameter:

Item	Payload type	code
Sensitivity	PRBS9	mChannelNumber=0x01; (change channel :0x14,0x26 mPayloadType=0x00; //PRBS9

(3)To setting Parameter with the Bluetooth test instrument.
(4)Run LE Packet Rx
(5)Stop LE Packet Rx and to obtain the receive packet count
(6) Calculate PER
If (PER< limit), go to step (3), else Stop

5.4.6 Verify Bluetooth BLE5.0 Rx Performance

To calculate the Packet Error Rate(PER)
PER%= 100*(1-(Packets Received /Packets Send)
Measure the DUT BLE Rx sensitivity to check Rx performance is ok or not. The Bluetooth Rx criterion is shown as below:

Channel	Spec	Payload Type	BLE sensitivity limit	Criterion
	PHY Spec			Bluetooth Spec

CH1-2404MHZ CH20-2442MHZ CH38-2478MHZ	1M	PRBS9	PER<= 30.800 %	-85 dBm
	2M	PRBS9	PER<= 30.800 %	-80 dBm
	LRS2	PRBS9	PER<= 30.800 %	-87 dBm
	LRS8	PRBS9	PER<= 30.800 %	-87 dBm

Table 20: The recommended test items of Bluetooth BLE 5.0 Rx

The simple sample code of RF dll show as:
The step by step list below:

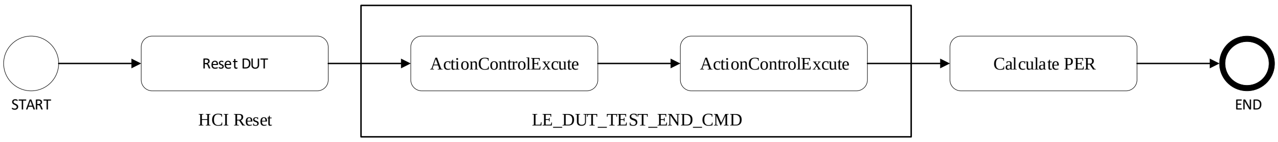


Figure 25: Verify Bluetooth BLE5.0 Rx

Enter Hci reset to reset the DUT.
Set Parameter & Run LE Packet Rx

Item	Packet type	

Sensitivity	LE2M	mChannelNumber=0x01; (change channel :0x14,0x26)
		mPayloadType=0x00; //PRBS9
		pParam->PHY = 2 ; // 1:LE 1M 2:LE 2M 3: LE Coded PHY
		pParam->ModulationIndex = 1

Sensitivity	LR	mChannelNumber=0x01; (change channel :0x14,0x26)
		mPayloadType=0x00; //PRBS9
		pParam->PHY = 3 ; // 1:LE 1M 2:LE 2M 3: LE Coded PHY
		pParam->ModulationIndex = 1

The LE RX sample code below:

```
pParam->ParameterIndex = LE_RX_DUT_TEST_CMD; //23
pParam->mChannelNumber= 1;                       //channel :1~38
pParam->PHY = 2 ;                // 1:LE 1M   2:LE 2M  3: LE Coded PHY
pParam->ModulationIndex = 1;  // 0: standard modulation 1: stable modulation
pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
pBluetoothModule->ActionControlExcute(pBluetoothModule
```

To setting Parameter with the Bluetooth test instrument ,then start TX until the end.
Stop LE Packet Rx and to obtain the receive packet count

```
pParam->ParameterIndex = LE_DUT_TEST_END_CMD; //24
pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
pBluetoothModule->ActionControlExcute(pBluetoothModule
```

Calculate PER:
If (PER< limit), go to step (2), else finish.

6 Write Data to Storage and Check Contents

Write board-dependent information into respective calibration data offset, this information include MAC address, calibrated TSSI DE, Thermal Meter, and so on.
Use the MP command:

```
iwpriv config_set wmap,offset,data
```

to send 16 bytes calibration data content to on-chip Calibration data area. If you want rewrite new contents, you can use above command again.

After above step, you should use below MP command to read partial contents,

```
iwpriv config_get rmap,offset,count
```

or use below MP command to get full contents

```
iwpriv config_get realmap
```

to check the written contents are all correct or not.

NOTE

Please see the appendix (12.) for detailed explanations of data contents.

6.1 WiFi OTP locations for MP calibration

Customers only need to fill on the following OTP locations for WiFi MP calibration. (The following default value may not be up to date, please contact Realtek FAE for the latest 'RTL8730E_default_OTP' file).

iwpriv config_set wmap,	0x020,	00	00	00	00	00	00	00	00	00	00	00	FF	FF	FF	FF	FF
iwpriv config_set wmap,	0x030,	FF	FF	00	00	00	00	00	00	00	00	00	00	00	00	00	00
iwpriv config_set wmap,	0x0C0,	FF	FF	FF	FF	FF	FF	FF	FF	FF	3F	66	FF	FF	FF	FF	FF
iwpriv config_set wmap,	0x110,	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	00	E0	4C	87	30	01
iwpriv config_set wmap,	0x130,	FF	21	FF	30	FF	FF	FF	FF	40	11	07	30	30	FF	FF	FF
iwpriv config_set wmap,	0x140,	FF	FF	FF	FF	00	FF	00	FF	00	FF	00	FF	00	FF	FF	FF

0x20~0x25	2.4G CCK TSSI DE
0x26~0x2A	2.4G BW20 TSSI DE
0x32~0x3F	5G BW20 TSSI DE
0xC9	Crystal Calibration
0xCA	Thermal meter
0x11A~0x11F	MAC Address
0x13B~0x13C	Country code
0x144/0x146	2.4G OFDM/CCK RX gain
0x148/0x14A/0x14C	5G Low/Middle/High Band RX gain

Table 21: WiFi OTP locations for MP calibration

6.2 BT OTP locations for MP calibration

For Bluetooth to write OTP, you have to use WiFi's API, cause of Bluetooth tool and API's is not support. (The following default value may not be up to date, please contact Realtek FAE for the latest 'RTL8730E_default_OTP' file).
The OTP define show as:
S0:

iwpriv config_set wmap,	0x1B0,	FF	FF	FF	FF	00	E0	4C	80	30	01	FF	FF	FF	FF	10	FF
iwpriv config_set wmap,	0x1C0,	FF	FF	FF	FF	53	04	FF	FF	4D	51	51	4D	4D	FF	FF	4D
iwpriv config_set wmap,	0x1D0,	FF	FF	FF	FF	64	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF

S1:

iwpriv config_set wmap,	0x1B0,	FF	FF	FF	FF	00	E0	4C	80	30	01	FF	FF	FF	FF	10	FF
iwpriv config_set wmap,	0x1C0,	FF	FF	FF	FF	52	04	FF	FF	4E	52	52	4E	4E	FF	FF	4E
iwpriv config_set wmap,	0x1D0,	FF	FF	FF	FF	64	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF

0x1B4 ~ 0x1B9	BD_ADDRESS
0x1BE[0]	Enable PHY Init[0], (default 0)
0x1BE[1]	Tx gain K valid bit[1]
0x1BE[2]	Flatness K valid bit[2]
0x1BE[4:3]	lbt_mode (default 0x2)
0x1BE[5]	lbt_enable (default 0)
0x1BE[7:6]	reserved (default 0)
0x1BF	TX Gain K
0x1C0 ~ 0x1C3	Flatness K
0x1C4	iqm_txgain_10dBm_raw_index(default s0:0x53, default s1:0x52)
0x1C5	lbt_ant_gain(default 0x4)
0x1C8	iqm_max_txgain_1M(default s0:0x4d, default s1:0x4e)
0x1C9	iqm_max_txgain_2M(default s0:0x51, , default s1:0x52)
0x1CA	iqm_max_txgain_3M(default s0:0x51, , default s1:0x52)
0x1CB	iqm_max_txgain_LE1M(default s0:0x4d, , default s1:0x4e)
0x1CC	iqm_max_txgain_LE2M (default s0:0x4d, , default s1:0x4e)
0x1CF	iqm_max_txgain_LR (default s0:0x4d, default s1:0x4e)
0x1D4	tmeterx4_txgaink_module(default 0x64)

Table 22: OTP define table

NOTE

[1]. If Tx gain K are been done please enable the valid bits.
[2]. If Flatness K are been done please enable the valid bits.

7 Appendix :BT Mass Production API's function Usage

This chapter briefly introduces how to link the dynamic link library to control DUT. The flow is shown as below:

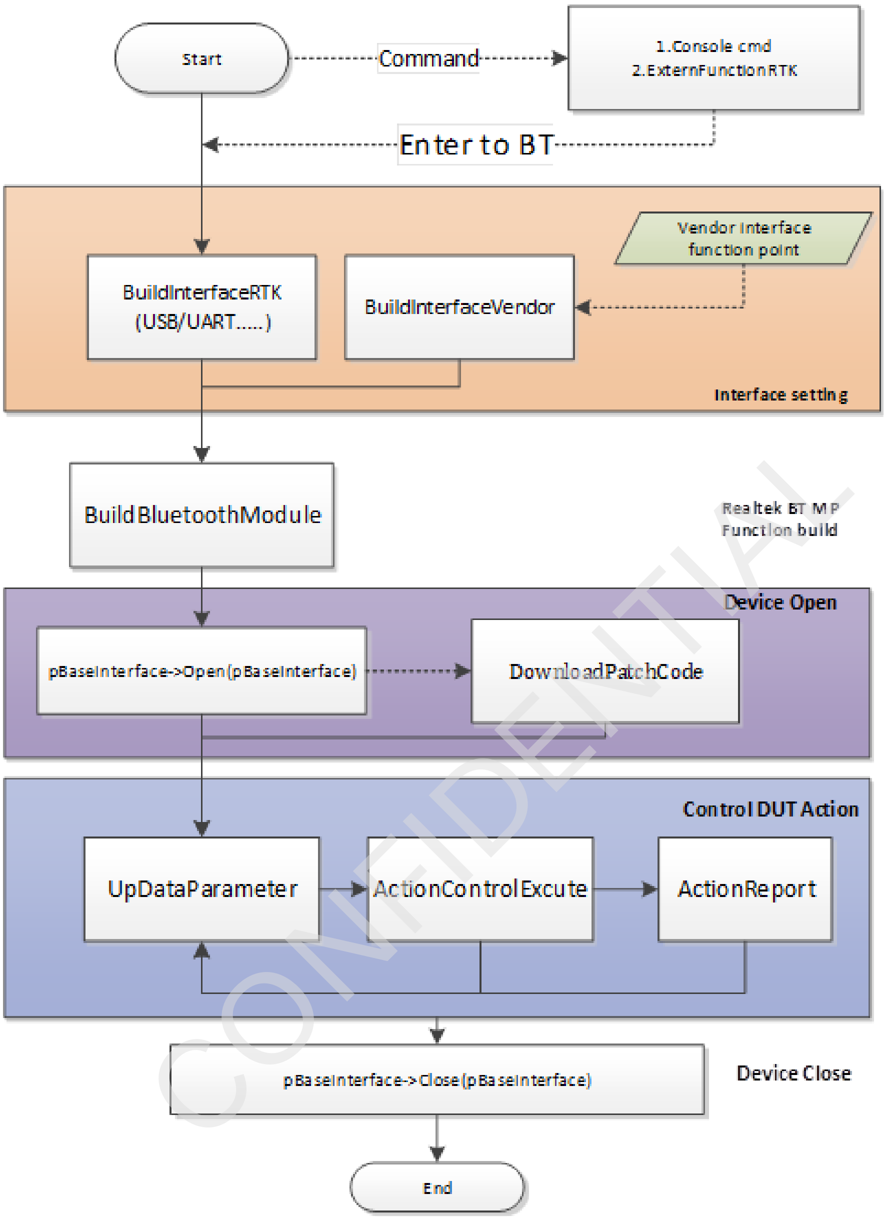


Figure 26: Mass Production API use flow

7.1 Use the API to enter BT HCI Mode: BTMPAPI_ExternFunctionRTK

BTMPAPI_BuildInterfaceVendor	
This function provides vender to implement how to open interface, close interface, send hci command, receive hci event, and wait mini-second functions by themselves.	
Syntax	
`int BTMPAPI_ExternFunctionRTK(int Chip_type, int Enter1_Leave2, BASE_INTERFACE_MODULE **ppBaseInterface, BASE_INTERFACE_MODULE *pBaseInterfaceModuleMemory, unsigned int InterfaceType, unsigned char PortNo, unsigned long Baudrate, unsigned char *pWriting, unsigned char *pReading, unsigned int return_check, unsigned char *pData, unsigned int Ending_close, unsigned int Delay);`	
Parameters	
Chip_type	Chip Numeber (Amebra Smart= 40)
Enter1_Leave2	Structure BASE_INTERFACE_MODULE point
**ppBaseInterface	Structure BASE_INTERFACE_MODULE double point
*pBaseInterfaceModuleMemory	Structure BASE_INTERFACE_MODULE point
InterfaceType,	Unsigned integer Value ,interface type
PortNo,	Unsigned integer Value,UART /USB Port number
Baudrate,	Unsigned integer Value, UART baudrate
*pWriting	Realtek define NULL
*pReading	Realtek define NULL
return_check	Realtek define NULL
*pData	Realtek define NULL
Ending_close	Realtek define 1
Delay	Delay time after send define cmd
Return Value	
Status	0: The operation completed successfully. Other:Error
Remakes	

```
int BTMPAPI_ExternFunctionRTK(
    int Chip_type,
    int Enter1_Leave2,
    BASE_INTERFACE_MODULE **ppBaseInterface,
    BASE_INTERFACE_MODULE *pBaseInterfaceModuleMemory,
    unsigned int   InterfaceType,
    unsigned char  PortNo,
    unsigned long  Baudrate,
    unsigned char *pWriting,
    unsigned char *pReading,
    unsigned int   return_check,
    unsigned char  *pData,
    unsigned int   Ending_close,
unsigned int   Delay
    );
```

Table 23: BTMPAPI_ ExternFunctionRTK API

7.2 BTMPAPI_BuildInterfaceVendor: Customer- interface

There are two options for the interface:
1. Customer-defined interface: The format of the defined interface API needs to be completed
2. Realtek has defined the interface
Therefore, you could select one between BuildInterfaceVendor and BuildInterfaceRTK.

For BTMPAPI_BuildInterfaceVendor API show as:

BTMPAPI_BuildInterfaceVendor	
This function provides vender to implement how to open interface, close interface, send hci command, receive hci event, and wait mini-second functions by themselves.	
Syntax	
`int BTMPAPI_BuildInterfaceVendor(BASE_INTERFACE_MODULE **ppBaseInterface, BASE_INTERFACE_MODULE *pBaseInterfaceModuleMemory, //Parmater unsigned int InterfaceType, unsigned char PortNo, unsigned long Baudrate, //basic fuction BASE_FP_OPEN Open, BASE_FP_SEND Send, BASE_FP_RECV Recv, BASE_FP_CLOSE Close, BASE_FP_WAIT_MS WaitMs);`	
Parameters	
**ppBaseInterface	Structure BASE_INTERFACE_MODULE double point
*pBaseInterfaceModuleMemory	Structure BASE_INTERFACE_MODULE point
InterfaceType,	Unsigned integer Value ,interface type
PortNo,	Unsigned integer Value,UART /USB Port number
Baudrate,	Unsigned integer Value, UART baudrate
Open	BASE_FP_OPEN function point
Send	BASE_FP_SEND function point
Recv	BASE_FP_RECV function point
Close	BASE_FP_CLOSE function point
WaitMs	BASE_FP_WAIT_MS function point
Return Value	
Status	0: The operation completed successfully. Other:Error
Remakes	
Base Funvtion point define:	

```
int BTMPAPI_BuildInterfaceVendor(
    BASE_INTERFACE_MODULE **ppBaseInterface,
    BASE_INTERFACE_MODULE *pBaseInterfaceModuleMemory,
    //Parmater
    unsigned int InterfaceType,
    unsigned char  PortNo,
    unsigned long Baudrate,
    //basic fuction
    BASE_FP_OPEN Open,
    BASE_FP_SEND Send,
    BASE_FP_RECV Recv,
    BASE_FP_CLOSE Close,
    BASE_FP_WAIT_MS WaitMs
    );
```

```
typedef int(*BASE_FP_OPEN)(BASE_INTERFACE_MODULE *pBaseInterface);

typedef int(*BASE_FP_SEND)(BASE_INTERFACE_MODULE *pBaseInterface,
unsigned char *pWritingBuf,
unsigned long Len);

typedef int(*BASE_FP_RECV )(BASE_INTERFACE_MODULE *pBaseInterface,
    unsigned char *pReadingBuf,
    unsigned long  Len,
    unsigned long *pRetLen
    );
typedef int(*BASE_FP_CLOSE)(BASE_INTERFACE_MODULE *pBaseInterface);
```

Table 24: BTMPAPI_BuildInterfaceVendor API

7.3 BTMPAPI_BuildInterfaceRTK: RTK's -interface

BTMPAPI_ BuildInterfaceRTK	
This function uses Realtek default functions to open, close, send, receive and wait functions.	
Syntax	
`int BTMPAPI_BuildInterfaceRTK( BASE_INTERFACE_MODULE **ppBaseInterface, BASE_INTERFACE_MODULE *pBaseInterfaceModuleMemory, unsigned int InterfaceType, unsigned char PortNo, unsigned long Baudrate );`	
Parameters	
**ppBaseInterface	Structure BASE_INTERFACE_MODULE double point
*pBaseInterfaceModuleMemory	Structure BASE_INTERFACE_MODULE point
InterfaceType,	Unsigned integer Value ,interface type
PortNo,	Unsigned integer Value,UART /USB Port number
Baudrate,	Unsigned integer Value, UART baudrate
Return Value	
Status	0: The operation completed successfully. Other: Error
Remakes	
None	

```
int BTMPAPI_BuildInterfaceRTK(
    BASE_INTERFACE_MODULE **ppBaseInterface,
    BASE_INTERFACE_MODULE *pBaseInterfaceModuleMemory,
    unsigned int InterfaceType,
    unsigned char   PortNo,
    unsigned long Baudrate
    );
```

Table 25: BTMPAPI_ BuildInterfaceRTK API

7.4 Realtek BT MP Function Build

BTMPAPI_ BuildInterfaceRTK	
This function to build mp function of Realtek in DLL.	
Syntax	
`typedef int (*pfBTMPAPI_BuildBluetoothModule)( BASE_BTMPDLL_MODULE *pBaseBTMPDLLModule, BASE_INTERFACE_MODULE *pBaseInterface, void *pExtra, unsigned char *pTxGainTable, unsigned char *pTxDACTable );`	
Parameters	
* pBaseBTMPDLLModule	Structure BASE_BTMPDLL_MODULE point
* pBaseInterface	Structure BASE_INTERFACE_MODULE point for interface
pExtra,	TBD, Realtek define is NULL
*pTxGainTable,	TBD, Realtek define is NULL
*pTxDACTable,	TBD, Realtek define is NULL
Return Value	
Status	0: The operation completed successfully. Other:Error
Remakes	
The MP function of Realtek Bluetooth device includes four main functions that are “DownloadPatchCode”, “UpDataParameter”, “ActionControlExcute”, and “ActionReport”.	

Table 26: BTMPAPI_ BuildInterfaceRTK API

7.5 Device Open

There is an action to open the device:
1. Open the interface.
2. Download patch code to device.

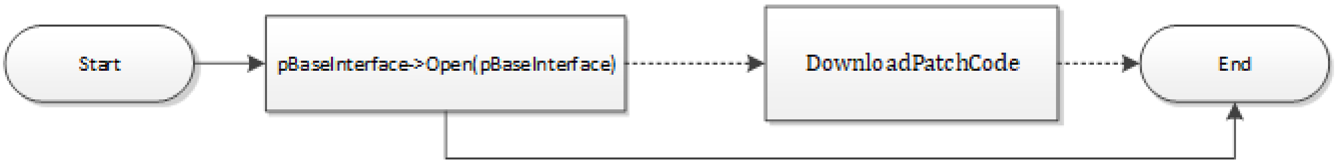


Figure 27: Device Open flow

7.5.1 Interface Open

Please use the open function in the data structure of the interface to open the interface.
As shown in the sample below:

```
pBaseInterface->Open(pBaseInterface);
```

7.5.2 DownloadPatchCode

DownloadPatchCode	
This function is used to download FW patch. Before downloading FW patch, vendor should read different patch files according to different BT chips.	
Syntax	
`typedef int(*BT_DLL_MODULE_FP_ACTION_DLFW)( BASE_BTMPDLL_MODULE *pDLLBtBaseModule, char *pPatchcode, int patchLength, int Mode );`	
Parameters	
* pDLLBtBaseModule	Structure BASE_BTMPDLL_MODULE point
* pPatchcode	Byte point, Array of Patchcode
patchLength,	Integer , Array length of Patchcode
Mode,	integer Value,0: Single 1: merge patch
Baudrate,	Unsigned integer Value, UART baudrate
Return Value	
Status	0: The operation completed successfully. Other:Error
Remakes	
None	

Table 27: DownloadPatchCode API

7.6 Control DUT Action

The Control function includes three main functions that are “UpDataParameter”, ActionControlExcute” and ActionReport”.

7.6.1 UpDataParameter

UpDataParameter
This function is to set parameters to DLL.
Syntax
`typedef int (*BT_DLL_MODULE_FP_UPDATA_PARAMETER)( BASE_BTMPDLL_MODULE *pDLLBtBaseModule, BT_PARAMETER *pParam`

);	
Parameters	
* pDLLBtBaseModule	Structure BASE_BTMPDLL_MODULE point
* pParam	Structure BT_PARAMETER point
Return Value	
Status	0: The operation completed successfully. Other: Error
Remakes	
None	

Table 28: UpDataParameter API

This function is used to pass "BT_PARAMETER" data structure to RtlBluetoothMP.dll. However, BT hardware registers don't be changed until "ActionControlExcute" is called. The following tables show more detail information about each parameter.

7.6.2 ActionControlExcute

ActionControlExcute	
Excute.	
Syntax	
`typedef int (*BT_DLL_MODULE_FP_ACTION_CONTROLEXCUTE)( BASE_BTMPDLL_MODULE *pDLLBtBaseModule );`	
Parameters	
* pDLLBtBaseModule	Structure BASE_BTMPDLL_MODULE point
Return Value	
Status	0: The operation completed successfully. Other: Error
Remakes	
None	

Table 29: ActionControlExcute API

According to "ParameterIndex" in the "BT_PARAMETER" structure, "ActionControlExecute" performs different functions, such as "PACKET_TX_START", "PACKET_TX_STOP", etc.

7.6.3 ActionReport

ActionReport	
Report to the execution result information.	
Syntax	
`typedef int (*BT_DLL_MODULE_FP_ACTION_REPORT)( BASE_BTMPDLL_MODULE *pDLLBtBaseModule, int ActiceItem, BT_DEVICE_REPORT *pReport );`	
Parameters	
* pDLLBtBaseModule	Structure BASE_BTMPDLL_MODULE point
ActiceItem	Integer, Report item.
*pReport	Structure BT_DEVICE_REPORT point.
Return Value	
Status	0: The operation completed successfully. Other: Error
Remakes	
None	

Table 30: ActionReport API

This function is used to report current TX/RX packet counts and chip status.The following table shows the relation between “ActionItem” and the parameters in “BT_DEVICE_REPORT”. For example, TotalTXBits and TotalTxCounts values are updated when “ActionItem” equls to “REPORT_PKT_TX”.

7.7 Device Close

Please use the open function in the data structure of the interface to open the interface. As shown in the sample below:

```
pBaseInterface->Close(pBaseInterface);
```

8 Appendix: BT Control API Usage

8.1 BT RF Direct Test Mode API's(DTM)

This chapter briefly introduces Direct test mode. If you want to know more details, please refer to the Bluetooth Spec.

8.1.1 BR/EDR–Direct Test Mode (Certification)

Enter the Bluetooth device to Bluetooth DUT Test and stop DUT Test Mode command below:
--Enter DUT Test Mode:

```
pParam->ParameterIndex = TEST_MODE_ENABLE;
pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
pBluetoothModule->ActionControlExcute(pBluetoothModule);
```

--Stop Test Mode:

```
pParam->ParameterIndex = HCI_RESET;
pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
pBluetoothModule->ActionControlExcute(pBluetoothModule) ;
Use bt_mp_Exec(HCI_RESET) to stop Bluetooth test mode.
```

The Test connection diagram:

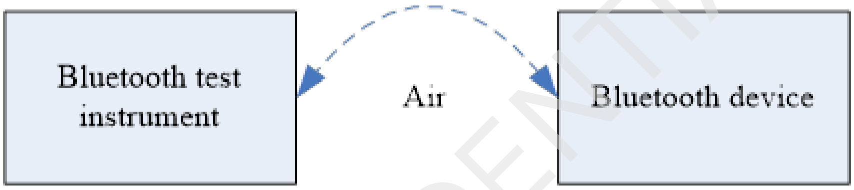


Figure 28 BR/EDR–Direct Test Mode

8.1.2 BLE Direct Test Mode (BQB Test)

After testing BT BLE only enable device and download patch code, you should jump interface to the instrument.

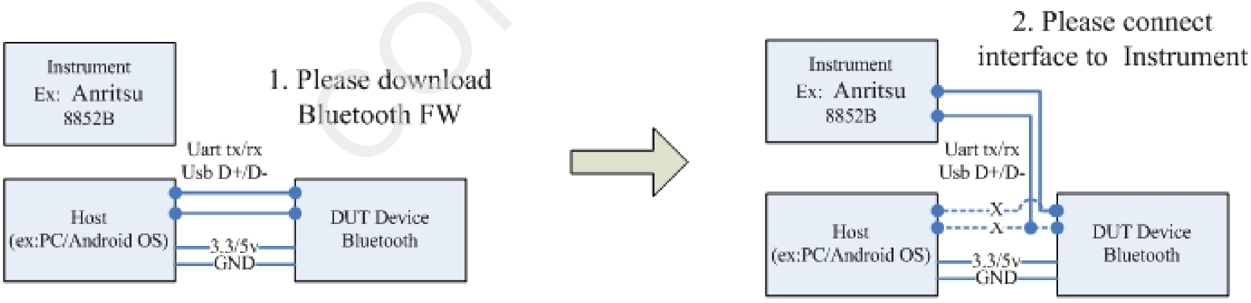


Figure 29: BLE Direct Test Mode

Step 1: Open Device
Step 2: Download Bluetooth Patch Code
Step 3: Jump hardware interface(UART) to the instrument.
Step 4: Begin test.

BLE is a schematic diagram of the test for Certification:

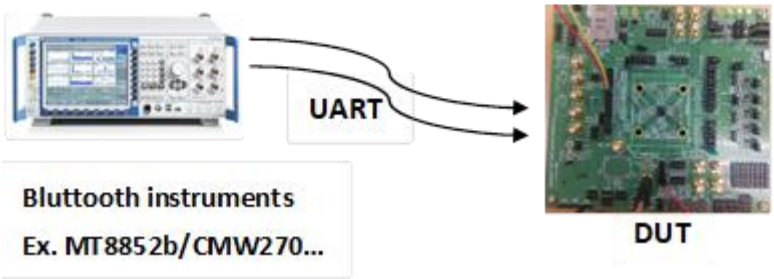


Figure 30: BLE Direct Test Mode connecter

8.1.3 LE DUT Test Mode-TX/RX (MP)

To start LE TX DUT test mode, follow the steps below:

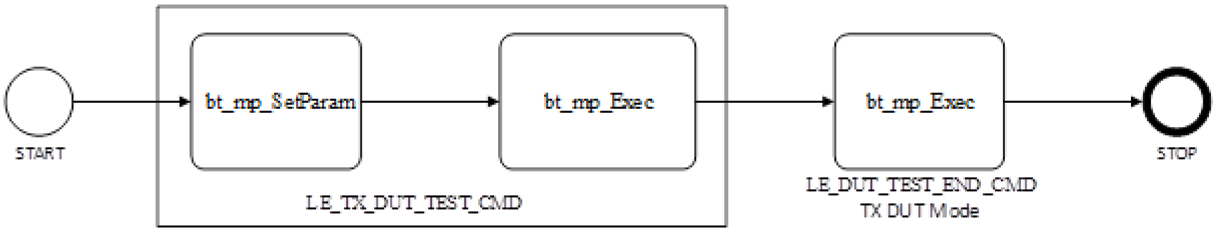


Figure 31: Start LE TX DUT test flow

Start LE RX DUT test mode, follow the steps below:

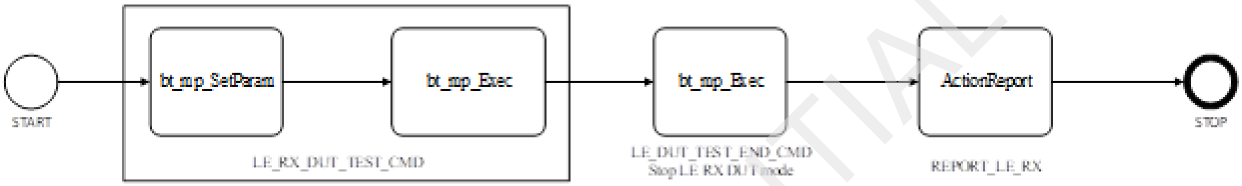


Figure 32: Start LE RX DUT test flow

To start LE TX DUT test mode, follow the steps below:

Step 1: bt_mp_SetParam

INDEX	VALUE	Value Range
1	ChannelNumber	1~38
3	PayloadType	BT_LE_PAYLOAD_TYPE_PRBS9 = 0, BT_LE_PAYLOAD_TYPE_1111_0000 = 1, BT_LE_PAYLOAD_TYPE_1010 = 2, BT_LE_PAYLOAD_TYPE_PRBS15 = 3, BT_LE_PAYLOAD_TYPE_ALL1 = 4, BT_LE_PAYLOAD_TYPE_ALL0 = 5, BT_LE_PAYLOAD_TYPE_0000_1111 = 6, BT_LE_PAYLOAD_TYPE_0101 = 7,
15	LEDataLen	0x00~0x25

Step 2: bt_mp_Exec(LE_TX_DUT_TEST_CMD)

```
pParam->ParameterIndex = LE_TX_DUT_TEST_CMD;
pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
pBluetoothModule->ActionControlExcute(pBluetoothModule);
```

Step 3: bt_mp_Exec(LE_DUT_TEST_END_CMD) to stop LE TX DUT mode.

```
pParam->ParameterIndex = LE_DUT_TEST_END_CMD;
pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
pBluetoothModule->ActionControlExcute(pBluetoothModule);
```

To start LE RX DUT test mode, follow the steps below:

Step 1: bt_mp_SetParam

INDEX	VALUE	Value Range
1	ChannelNumber	1~38

Step 2: bt_mp_Exec(LE_RX_DUT_TEST_CMD)

```
pParam->ParameterIndex = LE_RX_DUT_TEST_CMD;
pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
pBluetoothModule->ActionControlExcute(pBluetoothModule);
```

Step 3: bt_mp_Exec(LE_DUT_TEST_END_CMD) to stop LE RX DUT mode.

```
pParam->ParameterIndex = LE_DUT_TEST_END_CMD;
pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
pBluetoothModule->ActionControlExcute(pBluetoothModule);
```

Step 4: bt_mp_Report 11

```
//Update mBT_DEVICE_REPORT.TotalRxCounts
pBluetoothModule->ActionReport(pBluetoothModule,REPORT_LE_RX, &mBT_DEVICE_REPORT);
```

8.2 BT RF Test Mode API's(Certification)

This chapter describes the APIs of some operations of DUT during authentication.

8.2.1 TX Test -Continuous Tx(Certification)

In order to transmit a continuous signal, follow the process below:

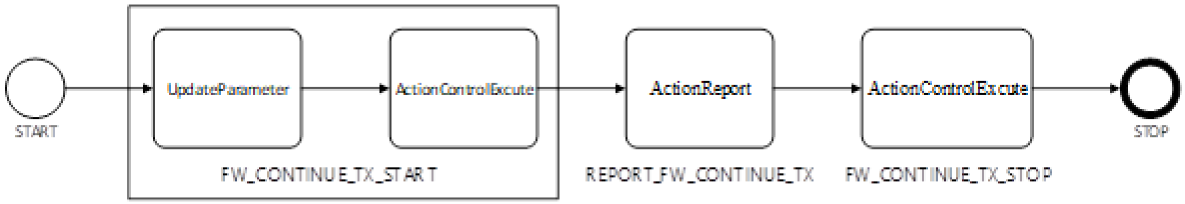


Figure 33: TX Test -Continuous Tx flow

Step1: UpDataParameter

Name
ParameterIndex
mChannelNumber
mPacketType
mPayloadType
mWhiteningCoeffValue
mTxGainIndex
mPacketHeader
mHitTarget

Step2: ActionControlExcute (CONTINUE_TX_START)

```
pParam->ParameterIndex = CONTINUE_TX_START;
pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
pBluetoothModule->ActionControlExcute(pBluetoothModule);
```

Step3: ActionReport (REPORT_CON_TX)

```
pBluetoothModule->ActionReport(pBluetoothModule,REPORT_CON_TX, pReport);
```

Step4: ActionControlExcute (CONTINUE_TX_STOP)

```
pParam->ParameterIndex = CONTINUE_TX_STOP;
pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
pBluetoothModule->ActionControlExcute(pBluetoothModule);
```

PS. "ActionReport" should be called every 200ms.

8.2.2 DUT Packet TX (MP)

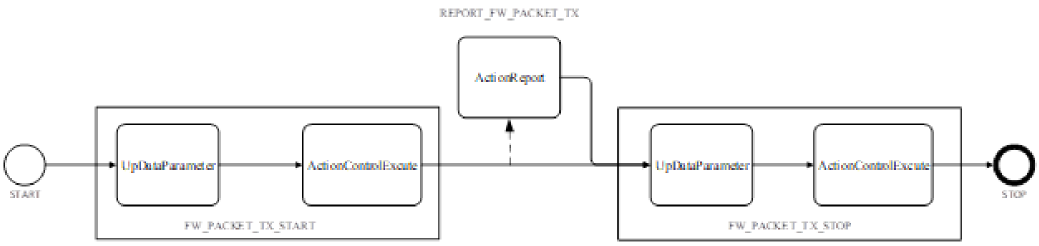


Figure 34: DUT Packet TX flow

In order to transmit a packet signal, follow the process below:
Step1: UpdataParameter

Name
ParameterIndex
mChannelNumber
mPacketType
mPayloadType
mTxPacketCount
mWhiteningCoeffValue
mTxGainIndex
mPacketHeader
mHitTarget

Step2: ActionControlExcute (PACKET_TX_START)

```
pParam->ParameterIndex = PACKET_TX_START;
pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
pBluetoothModule->ActionControlExcute(pBluetoothModule);
```

Step3: ActionReport (REPORT_PKT_TX)
pBluetoothModule->ActionReport(pBluetoothModule, REPORT_PKT_TX, pReport);

Step4: ActionControlExcute (PACKET_TX_STOP)

```
pParam->ParameterIndex = PACKET_TX_STOP;
pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
pBluetoothModule->ActionControlExcute(pBluetoothModule);
```

PS. "ActionReport " should be called every 200ms.

8.2.3 DUT Packet RX (MP)

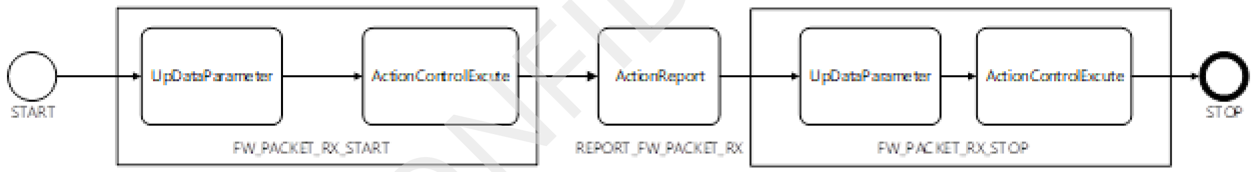


Figure 35: DUT Packet RX flow

In order to receive a packet signal, follow the process below:
Step1: UpDataParameter

Name
ParameterIndex
mChannelNumber
mPacketType
mPayloadType
mWhiteningCoeffValue
mPacketHeader
mHitTarget

Step2: ActionControlExcute (PACKET_RX_START)

```
pParam->ParameterIndex = PACKET_RX_START;
pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
pBluetoothModule->ActionControlExcute(pBluetoothModule);
```

Step3: ActionReport (REPORT_PKT_RX)

```
pBluetoothModule->ActionReport(pBluetoothModule, REPORT_PKT_RX, pReport);
```

Step4: ActionControlExcute (PACKET_RX_STOP)

```
pParam->ParameterIndex = PACKET_RX_STOP;
pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
```

```
pBluetoothModule->ActionControlExcute(pBluetoothModule);
```

PS. “ActionReport ” should be called every 200ms.

8.2.4 LE Certification TX Mode(Certification)

This function is only supported by some ICs .In order to facilitate RF safety testing, such as the band edge. We often use to facilitate continuous tx RF safety testing (eg.Bandedge).

Step1: UpDataParameter ; Channel Number = 1~38

INDEX	VALUE
1	ChannelNumber

Step2: ActionControlExcute (LE_TX_START_CMD)

Step4: ActionControlExcute (LE_TX_STOP_CMD)

8.2.5 Hopping Test (Certification- Dwell time)

In order to start hopping mode test, follow the steps below:
Step 1:
bt_mp_SetParam….(to setting packet type:BLE) and channel.

Name
PacketType
ChannelNumber

The hopping mode setting packet type define below:

Packet Type	INDEX
BT_PKT_LE	9
BT_PKT_NULL	10

Table 31 : Hopping mode packet type define

Step 2: bt_mp_Exec(HOPPING_DWELL_TIME)

```
pParam->ParameterIndex = HOPPING_DWELL_TIME;
pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
pBluetoothModule->ActionControlExcute(pBluetoothModule);
```

Step 3: bt_mp_Exec(HCI_RESET) to disable hopping mode.

```
pParam->ParameterIndex = HCI_RESET;
pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
pBluetoothModule->ActionControlExcute(pBluetoothModule) ;
```

9 Appendix: BT API's Define

9.1 INTERFACE_TYPE

Interface	Index	Note
USB	0	
UART	1	
WINUSB	4	

Table 32: Interface type index

```
The enum define is:
enum
{
TYPE_USB = 0,
    TYPE_UART,
    TYPE_FILTER_UART,
   TYPE_SDIO,
    TYPE_WINUSB,
    TYPE_ADB_UART,
    TYPE_ADB_USB,
    TYPE_SOCKET_DEVICE_UART,
    TYPE_SOCKET_DEVICE_USB,
    TYPE_BUMBLE_BEE_USB,

   TYPE_NUMBER
}INTERFACE_TYPE;
```

9.2 BT_ACTIONCONTROL_TAG

Command	Index	
HCI_RESET	0	▩
TEST_MODE_ENABLE	1	▩
REPORT_CLEAR	11	▩
HOPPING_DWELL_TIME	**21**	▩
LE_TX_DUT_TEST_CMD	**22**	▩
LE_RX_DUT_TEST_CMD	**23**	▩
LE_DUT_TEST_END_CMD	24	▩
FW_PACKET_TX_START	30	▩
FW_PACKET_TX_STOP	31	▩
FW_PACKET_RX_START	**32**	▩
FW_PACKET_RX_STOP	**33**	▩
FW_CONTINUE_TX_START	34	▩
FW_CONTINUE_TX_STOP	35	▩
FW_READ_TX_POWER_INFO	38	▩
SET_ANT_INFO	**40**	▩
TX_POWER_GAIN_K	45	▩
TX_POWER_FLATNESS	46	▩
TX_PATH_LOSS_MODULE	47	▩
SetTxPower_8822C	**51**	▩

Table 33: BT_ACTIONCONTROL_TAG

9.3 LE TX PHY

The LE Phy types are defined in Table LE PKT_TYPE:

BLE 5.0 Tx :NAME	INDEX
BLE5_TX_1M_PHY	1
BLE5_TX_2M_PHY	2
LE5_TX_CODED_PHY_S8	3
LE5_TX_CODED_PHY_S2	4
BLE 5.0 RX NAME	INDEX
BLE5_TX_1M_PHY	1
BLE5_TX_2M_PHY	2
LE5_TX_CODED_PHY_LR	3

Table 34: BLE TX PHY index

9.4 BLE_MODULATION_TYPE

NAME	INDEX	
STANDARD_MODULATION_INDEX	0	
STABLE_MODULATION_INDEX	1	

Table 35: BLE_MODULATION_TYPE

9.5 PKT_TYPE

The packet types are defined in Table PKT_TYPE:
Legacy:

NAME	INDEX	Payload Length in bits
BT_PKT_DH1	0	0~27*8
BT_PKT_DH3	1	0~183*8
BT_PKT_DH5	2	0~339*8
BT_PKT_2DH1	3	0~54*8
BT_PKT_2DH3	4	0~367*8
BT_PKT_2DH5	5	0~679*8
BT_PKT_3DH1	6	0~83*8
BT_PKT_3DH3	7	0~552*8
BT_PKT_3DH5	8	0~1021*8
BT_PKT_LE	9	0~39*8
BT_PKT_TYPE_NULL	10	None
BT_PKT_LE_2M	11	0~255
BT_PKT_LE_CODED_S8	12	0~255
BT_PKT_LE_CODED_S2	13	0~255

Table 36: PKT_TYPE for BDE/EDR and Hopping index

9.6 PAYLOAD_TYPE

The payload types are defined in Table PAYLOAD_TYPE.

NAME	INDEX
BT_LE_PAYLOAD_TYPE_PRBS9	0
BT_LE_PAYLOAD_TYPE_1111_0000	1
BT_LE_PAYLOAD_TYPE_1010	2
BT_LE_PAYLOAD_TYPE_PRBS15	3
BT_LE_PAYLOAD_TYPE_ALL1	4
BT_LE_PAYLOAD_TYPE_ALL0	5

Table 37: PAYLOAD_TYPE index for BLE

9.7 PACKET_HEADER

Packet type	Payload(Bits)	Packet Header Hex
DH1	216	33820
DH3	1464	39858
DH5	2712	A078
2DH1	432	33820
2DH3	2936	C050
2DH5	5432	3F870
3DH1	664	15C40
3DH3	4416	39858
3DH5	8168	A078

Table 38: PACKET_HEADER

9.8 Report ActiceItem

ActiceItem	Index	BT_DEVICE_REPORT
REPORT_PKT_TX	1	TotalTXBits TotalTXCounts
REPORT_CON_TX	2	TotalTXBits TotalTXCounts
REPORT_RKT_RX	3	RXRecvPktCnts TotalRXBits TotalRXCounts TotalRXErrorBits RXRssi ber Cfo
REPORT_TX_GAIN_TABLE	4	CurrTXGainTable
REPORT_TX_DAC_TABLE	5	CurrTXDACTable
REPORT_XTAL	6	CurrRtl8761Xtal
REPORT_CHIP	9	pBTInfo BTInfoMemory
REPORT_LE_RX	11	TotalRXCounts

Table 39: Report Actice Item index

10 Appendix: BT API's Structure

10.1 BASE_INTERFACE_MODULE

```
#define MAX_DATA_LEN    20

// Base interface module structure
struct BASE_INTERFACE_MODULE_TAG
{
    BASE_FP_OPEN Open;
    BASE_FP_SEND Send;
    BASE_FP_RECV Recv;
    BASE_FP_CLOSE Close;
    BASE_FP_WAIT_MS WaitMs;

    BASE_FP_SET_USER_DEFINED_DATA_POINTER   SetUserDefinedDataPointer;
    BASE_FP_GET_USER_DEFINED_DATA_POINTER   GetUserDefinedDataPointer;

    unsigned char InterfaceType;

    // User defined data
    unsigned long UserDefinedData;

    //for usb, uart
    unsigned char PortNo;

    //for uart
    unsigned long Baudrate;
    unsigned char bUartProtocol;

    unsigned char pData[MAX_DATA_LEN];
```

};

10.2 BT_PARAMETER

```
struct  BT_PARAMETER_TAG
{
    unsigned long ParameterIndex;

    unsigned char mPGRawData[MAX_USERAWDATA_SIZE];
    unsigned char mParamData[MAX_USERAWDATA_SIZE];
    unsigned char mChannelNumber;
    unsigned long mPacketType;
    unsigned char mTxGainIndex;
    unsigned char mTxGainValue;
    unsigned long mTxPacketCount;
    unsigned long   mPayloadType;
    unsigned long mPacketHeader;
    unsigned char mWhiteningCoeffValue;
    unsigned char mTxDAC;
    ULONG64 mHitTarget;

    unsigned char TXGainTable[MAX_TXGAIN_TABLE_SIZE];
    unsigned char TXDACTable[MAX_TXDAC_TABLE_SIZE];

    unsigned char bHoppingFixChannel;
    unsigned long Rtl8761Xtal;
    unsigned char ExeMode;
    unsigned char PHY;             //for Le Enhanced bt 5.0
    unsigned char ModulationIndex; //for Le Enhanced bt 5.0
};
```

Name	Value Range	Description
ParameterIndex	See BT_ACTION_CONTROL MAX Value	See BT_ACTION_CONTROL_TAG
mPGRawData	Raw data	
mChannelNumber	0~78	None
mPacketType	0~8	See PKT_TYPE
mTXGainIndex	0x00 (for default TX power)	
mTXPacketCount (for packet TX)	0~0x0FFFFFFF	0 : infinite TX packet count
mPayloadType	0~7	See BR/EDR PAYLOAD_TYPE and BLE PAYLOAD_TYPE
mPacketHeader	0x0~0x3FFFF	See PACKET_HEADER
mWhiteningCoeffValue	0x80 or 0x7F	0x7F : Enable Whitening 0x80 : Disable Whitening
mHitTarget	6 bytes	Instrument BD_ADDRESS
bHoppingFixChannel	0 : Disable 1 : Enable Fix Channel	For Hopping mode parameter
Rtl8761Xtal	0~0x7F (default 0x3F)	Crystal Cap Value

Table 40: UpDataParameter define: BT_PARAM

BT PKT_TYPE

NAME	INDEX	Payload Length in bits
BT_PKT_DH1	0	0~27*8
BT_PKT_DH3	1	0~183*8
BT_PKT_DH5	2	0~339*8
BT_PKT_2DH1	3	0~54*8
BT_PKT_2DH3	4	0~367*8
BT_PKT_2DH5	5	0~679*8
BT_PKT_3DH1	6	0~83*8
BT_PKT_3DH3	7	0~552*8
BT_PKT_3DH5	8	0~1021*8
BT_PKT_LE	9	0~39*8

Table 41: BT PKT_TYPE

BLE5.0_Tx_PKT_TYPE

NAME	INDEX
BLE5_TX_1M_PHY	1
BLE5_TX_2M_PHY	2
LE5_TX_CODED_PHY_S8	3
LE5_TX_CODED_PHY_S2	4

Table 42: BT BLE5.0_Tx_PKT_TYPE

BLE5.0_Rx_PKT_TYPE

NAME	INDEX
BLE5_TX_1M_PHY	1
BLE5_TX_2M_PHY	2
LE5_TX_CODED_PHY_LR	3

Table 43: BT BLE5.0_Rx_PKT_TYPE

BLE5.0_MODULATION_TYPE

NAME	INDEX
STANDARD_MODULATION_INDEX	0
STABLE_MODULATION_INDEX	1

Table 44: BT BLE5.0_MODULATION_TYPE

The payload types are defined in Table PAYLOAD_TYPE.

NAME	INDEX
BT_PAYLOAD_TYPE_ALL0	0
BT_PAYLOAD_TYPE_ALL1	1
BT_PAYLOAD_TYPE_0101	2
BT_PAYLOAD_TYPE_1010	3
BT_PAYLOAD_TYPE_0x0_0xF	4
BT_PAYLOAD_TYPE_0000_1111	5
BT_PAYLOAD_TYPE_1111_0000	6
BT_PAYLOAD_TYPE_PRBS9	7

Table 45: BT PAYLOAD_TYPE

LE_PAYLOAD_TYPE

NAME	INDEX
BT_LE_PAYLOAD_TYPE_PRBS9	0
BT_LE_PAYLOAD_TYPE_1111_0000	1
BT_LE_PAYLOAD_TYPE_1010	2
BT_LE_PAYLOAD_TYPE_PRBS15	3
BT_LE_PAYLOAD_TYPE_ALL1	4
BT_LE_PAYLOAD_TYPE_ALL0	5
BT_LE_PAYLOAD_TYPE_0000_1111	6
BT_LE_PAYLOAD_TYPE_0101	7

Table 46: BT LE_PAYLOAD_TYPE

PACKET TYPE

PACKET TYPE	PAYLOAD(Bits)	PACKET HEADER HEX
DH1	216	33820
DH3	1464	39858
DH5	2712	A078
2DH1	432	33820
2DH3	2936	C050
2DH5	5432	3F870
3DH1	664	15C40
3DH3	4416	39858
3DH5	8168	A078

Table 47: BT PACKET_HEADER

10.3 BASE_CAL_PARAMETER

```
#define MAX_BYTES_ARRARY_SIZE  4096
#define MAX_DOUBLE_ARRARY_SIZE 128

typedef struct BASE_CAL_ARRARY_TAG{
        unsigned Arrary8bit[MAX_BYTES_ARRARY_SIZE];
        double   ArraryDouble[MAX_DOUBLE_ARRARY_SIZE];
}BASE_CAL_ARRARY_TAG;

typedef struct BASE_CAL_RTK_TXPOWERK_PARAMETER_TAG{
        unsigned char TxGainKVlaue;
        unsigned int  TxFlatnessKValue;
}BASE_CAL_RTK_TXPOWERK_PARAMETER_TAG;

struct BASE_CAL_PARAMETER_TAG
{
            int Index;
            int chipID;
            BASE_CAL_ARRARY_TAG input;
            BASE_CAL_ARRARY_TAG output;
            BASE_CAL_RTK_TXPOWERK_PARAMETER_TAG txpowK;
            ////////////////////////////////////////////
```

```
            // Function point define
            BASE_FP_CAL_MEASUREMENT_CW fpMeasurementCW;
            BASE_FP_CAL_MEASUREMENT_TX fpMeasurementTX;
            void  *pUserdefinePoint;
            void  **ppUserdefineFuncPoint;

};
#endif  //end #ifndef BASE_CAL_DEFINE
```

11 Appendix: Tx Power Table

11.1 TX Gain K Define Table

The tx gain K mapping table show as:

TX gain K value (The unit is 0.5 dB per step.)	**OTP Tx Gain K Value**	**Value (2's complement)**
12	C	C
11	B	B
10	A	A
9	9	9
8	8	8
7	7	7
6	6	6
5	5	5
4	4	4
3	3	3
2	2	2
1	1	1
0	0	0
-1	FF	FF
-2	FE	FE
-3	FD	FD
-4	FC	FC
-5	FB	FB
-6	FA	FA
-7	F9	F9
-8	F8	F8
-9	F7	F7
-10	F6	F6
-11	F5	F5
-12	F4	F4

Table 48: TX Gain K Define Table

11.2 Flatness Define Table

TX Flatness value (The unit is 0.25 dB per step.)	**Flatness Value (2's complement)**	**OTP Flatness Value Byte Shift 2 bit (-->*4)**
12	0C	30
11	0B	2C
10	0A	28
9	9	24
8	8	20
7	7	1C
6	6	18
5	5	14
4	4	10
3	3	0C
2	2	8
1	1	4
0	0	0
-1	FF	FC
-2	FE	F8
-3	FD	F4
-4	FC	F0
-5	FB	EC
-6	FA	E8
-7	F9	E4
-8	F8	E0
-9	F7	DC
-10	F6	D8
-11	F5	D4
-12	F4	D0

Table 49: Flatness Define Table

11.3 Tx power Table

User can use Set power tx level API to set target Tx power.
Unit dBM
S0 TX Table

Tx power	GFSK/BLE power (dBm)	EDR config power (dBm)	Tx power	GFSK/BLE power (dBm)	EDR config power (dBm)
0x33	-6	-8	0x44	2.5	0.5
0x34	-5.5	-7.5	0x45	3	1
0x35	-5	-7	0x46	3.5	1.5
0x36	-4.5	-6.5	0x47	4	2
0x37	-4	-6	0x48	4.5	2.5
0x38	-3.5	-5.5	0x49	5	3
0x39	-3	-5	0x4A	5.5	3.5
0x3A	-2.5	-4.5	0x4B	6	4
0x3B	-2	-4	0x4C	6.5	4.5
0x3C	-1.5	-3.5	0x4D	7	5
0x3D	-1	-3	0x4E	7.5	5.5
0x3E	-0.5	-2.5	0x4F	8	6
0x3F	0	-2	0x50	8.5	6.5
0x40	0.5	-1.5	0x51	9	7
0x41	1	-1	0x52	9.5	7.5
0x42	1.5	-0.5	0x53	10	8
0x43	2	0			

Table 50: S0 Tx power Table

S1 TX Table

Tx power	GFSK/BLE power (dBm)	EDR config power (dBm)	Tx power	GFSK/BLE power (dBm)	EDR config power (dBm)
0x32	-6	-8	0x43	2.5	0.5
0x33	-5.5	-7.5	0x44	3	1
0x34	-5	-7	0x45	3.5	1.5
0x35	-4.5	-6.5	0x46	4	2
0x36	-4	-6	0x47	4.5	2.5
0x37	-3.5	-5.5	0x48	5	3
0x38	-3	-5	0x49	5.5	3.5
0x39	-2.5	-4.5	0x4A	6	4
0x3A	-2	-4	0x4B	6.5	4.5
0x3B	-1.5	-3.5	0x4C	7	5
0x3C	-1	-3	0x4D	7.5	5.5
0x3D	-0.5	-2.5	0x4E	8	6
0x3E	0	-2	0x4F	8.5	6.5
0x3F	0.5	-1.5	0x50	9	7
0x40	1	-1	0x51	9.5	7.5
0x41	1.5	-0.5	0x52	10	8
0x42	2	0			

Table 51: S1 Tx power Table

12 Appendix

12.1 WLAN and BT Controller Calibration Data Contents Spec

Bytes	Contents	Description	Value
20h	PathA/S0 2.4G CCK TSSI Offset	Bit[7] : Sign-bit for TSSI offset (0 for plus and 1 for minus.) Bit[6:0] : PathA 2.4G CCK TSSI offset for channel 1/2. The calculation is 2's complement. Ex : FF=-1 ,1=+1	00 h
21h		Bit[7] : Sign-bit for TSSI offset (0 for plus and 1 for minus.) Bit[6:0] : PathA 2.4G CCK TSSI offset for channel 3/4/5. The calculation is 2's complement. Ex : FF=-1 ,1=+1	00 h
22h		Bit[7] : Sign-bit for TSSI offset (0 for plus and 1 for minus.) Bit[6:0] : PathA 2.4G CCK TSSI offset for channel 6/7/8. The calculation is 2's complement. Ex : FF=-1 ,1=+1	00 h
23h		Bit[7] : Sign-bit for TSSI offset (0 for plus and 1 for minus.) Bit[6:0] : PathA 2.4G CCK TSSI offset for channel 9/10/11. The calculation is 2's complement. Ex : FF=-1 ,1=+1	00 h
24h		Bit[7] : Sign-bit for TSSI offset (0 for plus and 1 for minus.) Bit[6:0] : PathA 2.4G CCK TSSI offset for channel 12/13. The calculation is 2's complement. Ex : FF=-1 ,1=+1	00 h
25h		Bit[7] : Sign-bit for TSSI offset (0 for plus and 1 for minus.) Bit[6:0] : PathA 2.4G CCK TSSI offset for channel 14. The calculation is 2's complement. Ex : FF=-1 ,1=+1	00 h
26h	Path A/S0 2.4G BW20-1S TSSI Offset (Relative Value)	Bit[7] : Sign-bit for TSSI offset (0 for plus and 1 for minus.) Bit[6:0] : PathA 2.4G BW20-1S TSSI offset for channel 1/2. The calculation is 2's complement. Ex : FF=-1 ,1=+1	00 h
27h		Bit[7] : Sign-bit for TSSI offset (0 for plus and 1 for minus.) Bit[6:0] : PathA 2.4G BW20-1S TSSI offset for channel 3/4/5. The calculation is 2's complement. Ex : FF=-1 ,1=+1	00 h
28h		Bit[7] : Sign-bit for TSSI offset (0 for plus and 1 for minus.) Bit[6:0] : PathA 2.4G BW20-1S TSSI offset for channel 6/7/8. The calculation is 2's complement. Ex : FF=-1 ,1=+1	00 h
29h		Bit[7] : Sign-bit for TSSI offset (0 for plus and 1 for minus.) Bit[6:0] : PathA 2.4G BW20-1S TSSI offset for channel 9/10/11. The calculation is 2's complement. Ex : FF=-1 ,1=+1	00 h
2Ah		Bit[7] : Sign-bit for TSSI offset (0 for plus and 1 for minus.) Bit[6:0] : PathA 2.4G BW20-1S TSSI offset for channel 12/13/14. The calculation is 2's complement. Ex : FF=-1 ,1=+1	00 h
2Bh~31h	Reserved	Reserved for Realtek.	-
32h	PathA/S0 5G BW20-1S TSSI Offset (obtained from MP tool while performing power calibration)	"Bit[7] : Sign-bit for TSSI offset (0 for plus and 1 for minus.) Bit[6:0] : PathA 5G BW20-1S TSSI offset for channel 36/38/40. The calculation is 2's complement. Ex : FF=-1 ,1=+1"	00h
33h		"Bit[7] : Sign-bit for TSSI offset (0 for plus and 1 for minus.) Bit[6:0] : PathA 5G BW20-1S TSSI offset for channel 44/46/48. The calculation is 2's complement. Ex : FF=-1 ,1=+1"	00h
34h		"Bit[7] : Sign-bit for TSSI offset (0 for plus and 1 for minus.) Bit[6:0] : PathA 5G BW20-1S TSSI offset for channel 52/54/56. The calculation is 2's complement. Ex : FF=-1 ,1=+1"	00h
35h		"Bit[7] : Sign-bit for TSSI offset (0 for plus and 1 for minus.) Bit[6:0] : PathA 5G BW20-1S TSSI offset for channel 60/62/64. The calculation is 2's complement. Ex : FF=-1 ,1=+1"	00h
36h		"Bit[7] : Sign-bit for TSSI offset (0 for plus and 1 for minus.) Bit[6:0] : PathA 5G BW20-1S TSSI offset for channel 100/102/104. The calculation is 2's complement. Ex : FF=-1 ,1=+1"	00h
37h		Bit[7] : Sign-bit for TSSI offset (0 for plus and 1 for minus.)	00h

		Bit[6:0] : PathA 5G BW20-1S TSSI offset for channel 108/110/112. The calculation is 2's complement. Ex : FF=-1 ,1=+1	
38h		Bit[7] : Sign-bit for TSSI offset (0 for plus and 1 for minus.) Bit[6:0] : PathA 5G BW20-1S TSSI offset for channel 116/118/120. The calculation is 2's complement. Ex : FF=-1 ,1=+1	00h
39h		Bit[7] : Sign-bit for TSSI offset (0 for plus and 1 for minus.) Bit[6:0] : PathA 5G BW20-1S TSSI offset for channel 124/126/128. The calculation is 2's complement. Ex : FF=-1 ,1=+1	00h
3Ah		Bit[7] : Sign-bit for TSSI offset (0 for plus and 1 for minus.) Bit[6:0] : PathA 5G BW20-1S TSSI offset for channel 132/134/136. The calculation is 2's complement. Ex : FF=-1 ,1=+1	00h
3Bh		Bit[7] : Sign-bit for TSSI offset (0 for plus and 1 for minus.) Bit[6:0] : PathA 5G BW20-1S TSSI offset for channel 140/142/144. The calculation is 2's complement. Ex : FF=-1 ,1=+1	00h
3Ch		Bit[7] : Sign-bit for TSSI offset (0 for plus and 1 for minus.) Bit[6:0] : PathA 5G BW20-1S TSSI offset for channel 149/151/153. The calculation is 2's complement. Ex : FF=-1 ,1=+1	00h
3Dh		Bit[7] : Sign-bit for TSSI offset (0 for plus and 1 for minus.) Bit[6:0] : PathA 5G BW20-1S TSSI offset for channel 157/159/161. The calculation is 2's complement. Ex : FF=-1 ,1=+1	00h
3Eh		Bit[7] : Sign-bit for TSSI offset (0 for plus and 1 for minus.) Bit[6:0] : PathA 5G BW20-1S TSSI offset for channel 165/167/169. The calculation is 2's complement. Ex : FF=-1 ,1=+1	00h
3Fh		Bit[7] : Sign-bit for TSSI offset (0 for plus and 1 for minus.) Bit[6:0] : PathA 5G BW20-1S TSSI offset for channel 173/175/177. The calculation is 2's complement. Ex : FF=-1 ,1=+1	00h
40h~C7h	Reserved	Reserved for Realtek.	-
C8h	Channel Plan	Please set Country Code (0x13B 0x13C) to select channel plan	FF h
C9h	Crystal Calibration	XTAL_K Value Bit[6:0], Xi=Xo Range 0~7F h. (If in the case without assigned value, 0x3F will be used by driver.) Bit[7]: reserved	3F h
CAh	Thermal Meter	Thermal Meter Default Value System maker will calibrate a value and save it in EEPROM. Bit[7:0]: Thermal Meter Value 0xFF : Disable Tx power tracking function	80 h
CBh	IQ Calibration and LC Calibration	Reserved	00 h
CCh	2G and 5G PA Type	Reserved	0F h
CDh	2G LNA Type and Gain Selection	Reserved	00 h
CEh		Reserved	00 h
CFh	5G LNA Type and Gain Selection	Reserved	FF h
130h		Reserved	FF h
11Ah~0x11Fh	Function 1 CIS Data MAC Address	MAC Address: After the auto-load command or hardware reset, the Ameba-series chip loads MAC Addresses to MACID of the I/O registers of the Ameba-series chip.	00E04C873001h
131h	Board Options	Bit[1:0]: Regulatory selection. 0h: The Tx output power varies according to power-by-rate table. (Power limit table take no effect on output power) (non-regular use)	21 h

		1h: The Tx output power varies according to both of power-by-rate table and power-limit-table. (Power-by-rate table is generally provided by RTK according to optimized through-put test.) (Power limit table is based on certification-pass criterion.) (Default setting to satisfy both of the two purposed.) 2h: Disable power-by-rate function. 3h: reserved Bit[2]: Tx diversity 0: disable or multi-Tx-path 1: enable (de-featured to 1 Tx/stream) Bit[3]: Rx diversity 0: disable or multi-Rx-path (Rx MRC) 1: enable (de-featured to 1 Rx/stream) Bit[4]: option for 11ac mode 0: enable 11ac mode 1: disable 11ac mode Bit[7:5]: Module type 0h: WiFi solo module 1h: WiFi/BT combo module 2~7h: reserved	
132h	Feature Options	Bit[1:0]: function configuration of pin_LED0 and pin_LED1 Bit[3:2]: Link Speed shown in OS 0h: Current Tx PHY Rate 1h: Current Rx PHY Rate 2h: Maximum RX PHY Rate 3h: reserved Bit[4]: WLAN disable mode selection 0: radio off 1: power down Bit[7:5]: reserved	00 h
133h	BT Setting (from BT's point of view, not concerned with WiFi)	Bit[0]: Total antenna number 0: BT/WiFi use dedicated antenna 1: BT/WiFi use shared antenna Bit[3:1]: Co-existence type 0h: Internal Mailbox (default) 1h: RTK BT I2C Mailbox (contact FAE for more information) Bit[4]: antenna isolation 0: High isolation (>35dB) 1: Low (default) Bit[5]: Radio on/off type 0: combine with WiFi/BT SEL Shared Port 1:individual /BT SEL Dedicated Port Bit[7:6]: reserved	11 h
134h	Version	The EEPROM content version	00h
135h	Customer ID	Customer ID (0x00 and 0xFF are reserved for Realtek)	00h
136h	2G Tx BB Swing Setting	Bit[1:0]: 2G PathA OFDM 0h: 0dB (default), 1h: -3dB, 2h: -6dB, 3h: -9dB	00h
137h	5G Tx BB Swing Setting	Bit[1:0]: 5G Path A OFDM 0h: 0dB (default)	00h
138h	Tx Power Calibration Rate & Tx Power Tracking Mode	Power Calibration Bit[0]: 2G 40M Tx Power Calibrator Rate. 0h : HT40, MCS7 64QAM (default); 1h : VHT40, MCS9 256QAM Bit[1]: 5G 40M Tx Power Calibrator Rate. 0h : HT40, MCS7 64QAM (default); 1h : VHT40, MCS9 256QAM	40h

		Bit[3:2]: reserved Power Tracking Bit[7:4] 0x0h ~ 0x3h for power tracking by thermal; 0x4h ~ 0x7h for power tracking by TSSI	
139h	T/Rx Path Selection	Bit[7:4] Tx path assignment Bit[7]: path S3; Bit[6]: path S2; Bit[5]: path S1; Bit[4]: path S0; (1 for on and 0 for off) Bit[3:0] Rx path assignment Bit[3]: path S3; Bit[2]: path S2; Bit[1]: path S1; Bit[0]: path S0; (1 for on and 0 for off) (Remark: The ability for maximum Tx/Rx paths would be constrained by real physical limitation.)	11h
13Ah	RFE Type (RF Front-end Type) (Concerned with hardware parameter.)	Bit[7]: Determined RFE type selection by driver or OTP 1h: RFE type defined by driver. (discard the below setting of 0x13A[6:0]) 0h: RFE type defined by 0x13A[6:0]. Bit[6:0]: RFE Type # Please contact FAE to get right RFE type for individual hardware design.	01h
13Bh	Country code	ISO 3166-1 two-letter country code. Driver will set channel plan and power limit table according to this country code. SDK version 'ameba_all_V1.0' or later supports this function. Ex.1 : country code 'CN' for China according to ISO 3166-1, then write 0x13B = 0x43 , ASCII code for 'C' 0x13C = 0x4E , ASCII code for 'N' Ex.2: 0x13B/0x13C=0x30/0x30='0'/'0' for wordwide ACSII chars table：https://www.asciicharstable.com/	30h
13Ch	Country code		30h
13Dh~143h	Reserved	Reserved for Realtek.	-
144h	RxGainGap_2GOFDM	2G Rx gain calibration (OFDM) Bit[7:4] : for main path Bit[3:0] : for aux path 2's Compensation representation Ex：0x3h = +3dB, 0xD = -3dB	00h
145h	Reserved for Realtek.	Reserved for Realtek.	-
146h	RxGainGap_2GCCK	2G Rx gain calibration (CCK) Bit[7:4] : for main path Bit[3:0] : for aux path 2's Compensation representation Ex：0x3h = +3dB, 0xD = -3dB	00h
147h	Reserved for Realtek.	Reserved for Realtek.	-
148h	RxGainGap_5GL	5G Rx gain calibration (Low band) Bit[7:4] : for main path Bit[3:0] : for aux path Compensation representation 2's Compensation representation Ex：0x3h = +3dB, 0xD = -3dB	00h
149h	Reserved for Realtek.	Reserved for Realtek.	-
14Ah	RxGainGap_5GM	5G Rx gain calibration (Middle band) Bit[7:4] : for main path Bit[3:0] : for aux path 2's Compensation representation Ex：0x3h = +3dB, 0xD = -3dB	00h
14Bh	Reserved for Realtek.	Reserved for Realtek.	-
14Ch	RxGainGap_5GH	5G Rx gain calibration (High band) Bit[7:4] : for main path Bit[3:0] : for aux path 2's Compensation representation Ex：0x3h = +3dB, 0xD = -3dB	00h
14Dh	Reserved for Realtek.	Reserved for Realtek.	-
1B0h~1B3h	Reserved	Reserved for Realtek. Do not change this field without Realtek's	-

		approval.	

1B4h~1B9h	BD_ADDRESS	BD_ADDRESS
		//example: PG bd_addr = 0x00E04C803001:
		0x1B4=00
		0x1B5=E0
		0x1B6=4C
		0x1B7=80
		0x1B8=30
		0x1B9=01
			00E04C803001h

1BEh	Function Enable	Bit[0]:Enable PHY Init[0], (default 0)
		Bit[1]: Tx gain K valid bit *
		Bit[2]: Flatness K valid bit **
		Bit[4:3]: lbt_mode (default 2)
		Bit[5]: lbt_enable (default 0)
		Bit[7:6]: reserved (default 0)
		Note:
		//After Tx gain K flow,
		Please remember to enable 0x1BE will be 0x12.
		//After Flatness K flow,
		Please remember to enable 0x1BE will be 0x14.
		//After Tx gain K and Flatness K flow,
		Please remember to enable 0x1BE will be 0x16.
			10h

1BFh	TX Gain K	Bit[7:0]: TX Gain K	00h
1C0h~1C3h	TX Flatness K	Bit[7:0]: Flatness K	00h
1C4h	iqm_txgain_10dBm_raw_index	iqm_txgain_10dBm_raw_index (default s0:0x53, default s1:0x52)	S0:53h
1C5h	lbt_ant_gain	Bit[7:0]: lbt_ant_gain(default 0x4)	04h
1C6h~1C7h	Reserved	Reserved for Realtek. Do not change this field without Realtek's approval.	-
1C8h	iqm_max_txgain_1M	1M Max Tx Gain //Note: default s0:0x4d(7dBm), default s1:0x4e(8dbm)	S0:4Dh
1C9h	iqm_max_txgain_2M	2M Max Tx Gain //Note: default s0:0x51(7dBm), default s1:0x52(8dbm)	S0:51h
1CAh	iqm_max_txgain_3M	3M Max Tx Gain //Note: default s0:0x51(7dBm), default s1:0x52(8dbm)	S0:51h
1CBh	iqm_max_txgain_LE1M	LE1M Max Tx Gain //Note: default s0:0x4d(7dBm), default s1:0x4e(8dbm)	S0:4Dh
1CCh	iqm_max_txgain_LE2M	LE2M Max Tx Gain //Note: default s0:0x4d(7dBm), default s1:0x4e(8dbm)	S0:4Dh
1CDh~1CEh	Reserved	Reserved for Realtek. Do not change this field without Realtek's approval.	-
1CFh	iqm_max_txgain_LR	LR Max Tx Gain //Note: default s0:0x4d(7dBm), default s1:0x4e(8dbm)	S0:4Dh
1D0h~1D3h	Reserved	Reserved for Realtek. Do not change this field without Realtek's approval.	-
1D4h	tmeterx4_txgaink_module	T meter value	64h
1D5h~1FFh	Reserved	Reserved for Realtek. Do not change this field without Realtek's approval.	-

Table 52: WLAN and BT Controller Calibration Data Contents

12.2 Appendix: Hardware Tx parameters

```
iwpriv mp_hwtx period=[Packet Interval],len=[Packet Length],count=[Packet Count]
```

[Packet Interval]	(Option) 1~65535us, default is 100(unit：1us)
[Packet Length]	(Option) Data length of packet payload, default is 1500
[Packet Count]	(Option) Set the number of packets to send. 0 is for continuous packet Tx.

12.3 Appendix: ASCII characters table

Dec	Hex	Oct	Chr	Dec	Hex	Oct	HTML	Chr	Dec	Hex	Oct	HTML	Chr	Dec	Hex	Oct	HTML	Chr
0	0	000	NULL	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	Start of Header	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	Start of Text	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	End of Text	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	End of Transmission	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	Enquiry	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	Acknowledgment	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	Bell	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	Backspace	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	Horizontal Tab	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	Line feed	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	Vertical Tab	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	Form feed	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	Carriage return	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	Shift Out	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	Shift In	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	Data Link Escape	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	Device Control 1	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	Device Control 2	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	Device Control 3	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	Device Control 4	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	Negative Ack.	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	Synchronous idle	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	End of Trans. Block	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	Cancel	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	End of Medium	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	Substitute	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	Escape	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	File Separator	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	Group Separator	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	Record Separator	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	Unit Separator	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	Del