RTL8721DA/RTL8721DC/RTL8721DG
MP FLOW

Version 1.10

COPYRIGHT

DISCLAIMER

TRADEMARKS

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	2023/09/26	First Release.	Winnita, Tim
2.0	2024/01/04	Add 1.8V & 3.3V DC power supply mode.	Winnita, Tim
3.0	2024/01/12	Add command sample for WiFi calibration and verification.	Winnita, Tim
4.0	2024/03/06	Modify 3.1.1.2 DC power supply mode. Modify 12. Appendix 1BE description. Modify 5.3.2 and 5.3.3 flow and sample code Modify 5.4.1 and 5.4.3 content for BLE 1M TX/RX Add 9.6 Action control index table	Winnita, Tim, Dean
5.0	2024/03/29	Modify description for Calibration Data Contents.	Winnita, Tim, Dean
6.0	2024/05/20	Modify Appendix/OTP_map for country code setup.	Winnita, Tim, Dean
7.0	2024/05/24	Model name Correction.	Winnita, Tim, Dean
8.0	2024/09/30	Modify BT OTP default value. Add 2-antenna TRX performance verification.	Winnita, Tim, Dean
9.0	2024/11/13	1. Modify WL TX CCK calibration method. 2. Modify BT Tx power Table and criterion 1) 5.4.1 and 5.4.2 criterion 2) 11.3 Tx power table	Winnita, Tim, Dean
10.0	2025/03/20	Fix table 6-1 WiFi OTP locations	Winnita, Tim, Dean

Table of Contents

List of Tables

List of Figure

1 Test platform

The calibration flow described in following sections is 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:

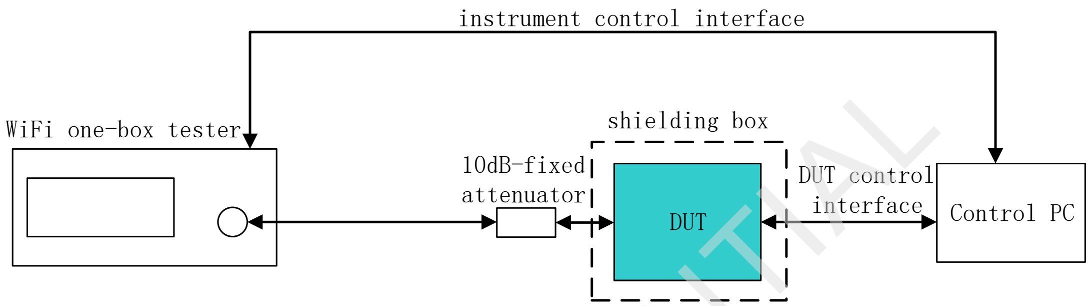


Figure 1: Brief diagram about test environment setup of 1-antenna DUT

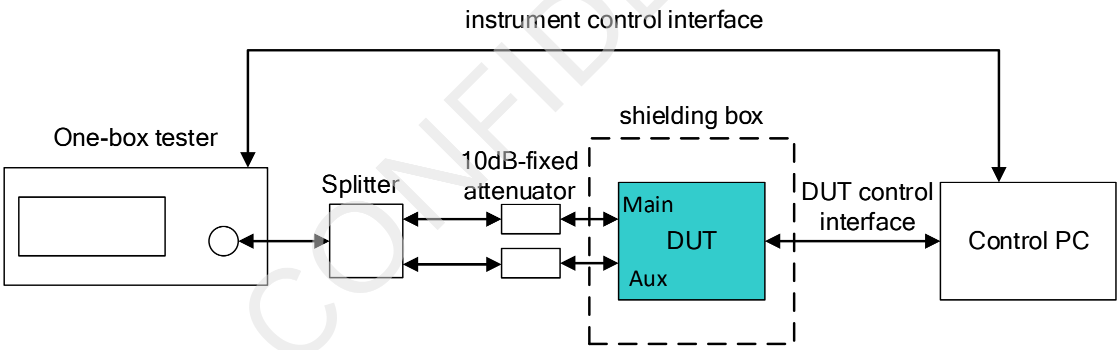


Figure 2: Brief diagram about test environment setup of 2-antenna DUT

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.

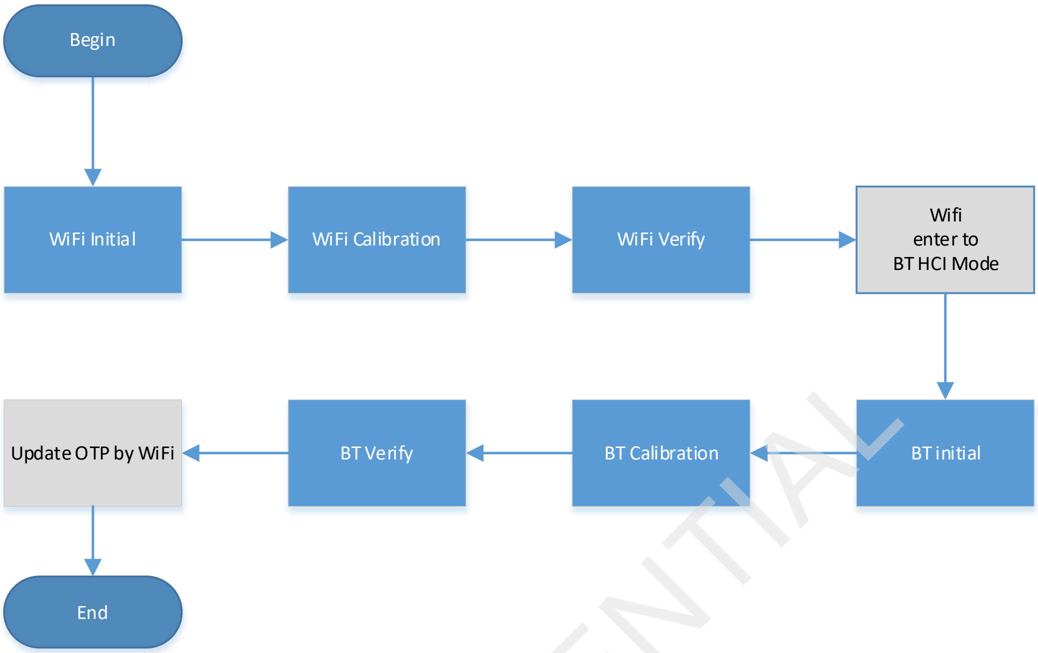


Figure 3: DUT MP Flow

3 WiFi Mass Production Flow

3.1 WiFi calibration

The WiFi calibration steps are shown as Figure 3:

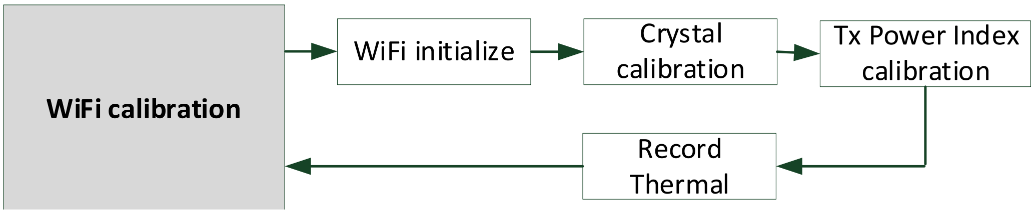


Figure 4: WiFi calibration flow

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_start
iwpriv mp_stop
```

```
#iwpriv mp_start

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

For different DC power supply modes, the driver will load different RF parameters for optimized performance. So, before you enter MP mode, you should send command to tell driver which mode you will select.

- Power Supply Mode 0, system DC power = 3.3V, PA DC power = 3.3V
  - In this default mode, there is no need to send extra command before you enter MP mode
- Power Supply Mode 1, system DC power = 1.8V, PA DC power = 1.8V
  - You should send extra command before you enter MP mode, as follows

```
iwpriv pwr_supply_mode 1
iwpriv mp_start
```

- Power Supply Mode 2, system DC power = 3.3V, PA DC power = 1.25V
  - You should send extra command before you enter MP mode, as follows

```
iwpriv pwr_supply_mode 2
iwpriv mp_start
```

NOTE

Please refer to 'RTL8721DA_RTL8721DC_RTL8721DG_RF DC Power Supply Modes Application Guide.pdf' for more details.

If the DUT has two antennas with diversity, you should add command to tell driver which rfe_type the DUT is. Please send rfe command before you enter MP mode.

```
iwpriv mp_rfe 2          // Set rfe type, default value is 2.
iwpriv mp_start
```

Then you can use the following command to select antenna.

```
iwpriv mp_antsw 1        //Select 1/2 TRX antenna(according to rfe_type),default is 1.
```

NOTE

Please refer to 'RTL8721DA_RTL8721DC_RTL8721DG-RFE Type Definition' *for more details.*
If there is any concern, please contact Realtek FAE.

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_start          //Enter WiFi MP mode
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_rate 108
iwpriv mp_txpower patha=64,pathb=0
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_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_rate 108
iwpriv mp_txpower patha=64,pathb=0
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 3-1: Crystal calibration offset in OTP

3.1.2.2 Calibrated Crystal Cap. Flow

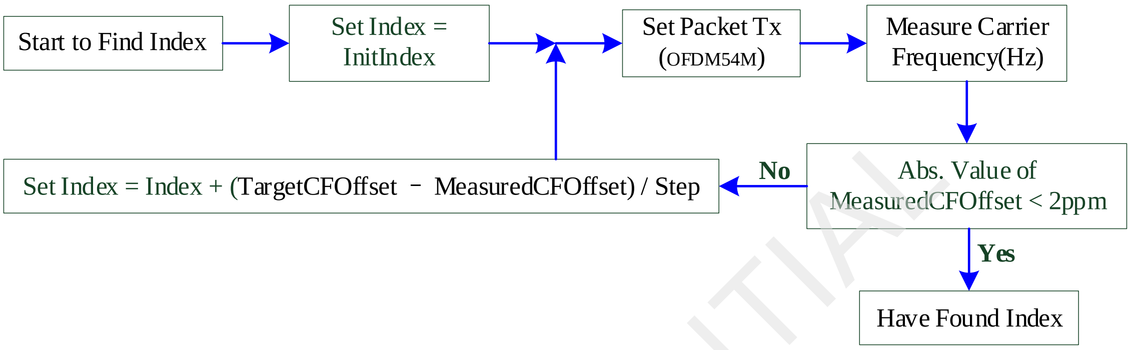


Figure 5: Finding Crystal Cap. Index flow

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 patha=64,pathb=0
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 WiFi Calibrates Tx Power Index

3.1.3.1 OTP definition about Tx power index and thermal meter

First, take a look at OTP content about setting of RF Tx gain index. Normal driver will load bellow Tx gain setting for each channel group or each PHY data rate. So these Tx gain setting must be well-calibrated and filled on correct OTP location.

2.4Ghz 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 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]		

Power Difference Location in OTP		
2G Band	54M-1T to MCS7-B40	0x2B[3:0]
	MCS7-B20 to MCS7-B40	0x2B[7:4]
5G Band	54M-1T to MCS7-B40	0x40[3:0]
	MCS7-B20 to MCS7-B40	0x40[7:4]

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

Table 3-2: Tx gain index 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		**MCS7-B40**	**MCS7-B20**	**54M**	**CCK**
2.4G Target Power	3.3V power supply	17dBm	17dBm	18dBm	20dBm
	1.8V power supply	11dBm	11dBm	12dBm	15dBm
5G Target Power	3.3V power supply	15dBm	15dBm	16dBm	
	1.8V power supply	8dBm	8dBm	9dBm	

Table 3-3: 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 Tx calibration flow

Theoretically, we need to measure all value defined above in OTP to calibrate the Tx power level. But since it needs too much time, we only measure several channels with MCS7-B40 signal and CCK-11M signal, and figure out the other non-measured value by some easy mathematics method.
Usually, the recommended measured channels are listed below:

2G Band(CCK)			2G Band(MCS7)		5G Band1		5G Band2		5G Band3		5G Band4	
CH4	CH10	CH14	CH4	CH10	CH38	CH46	CH54	CH62	CH102	CH142	CH151	CH175

Table 3-4: The recommended measured channel for Tx power calibration

The flow is shown as below:

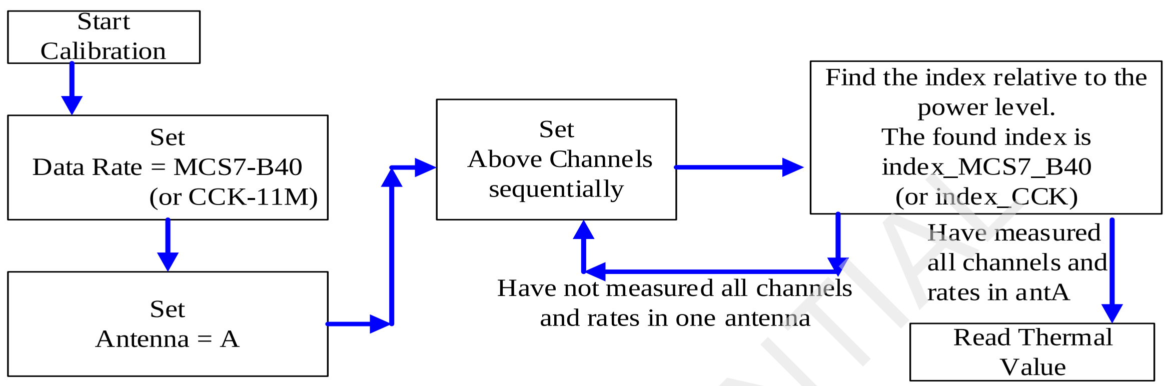


Figure 6: Tx calibration flow

Each finding index flow is shown as below:

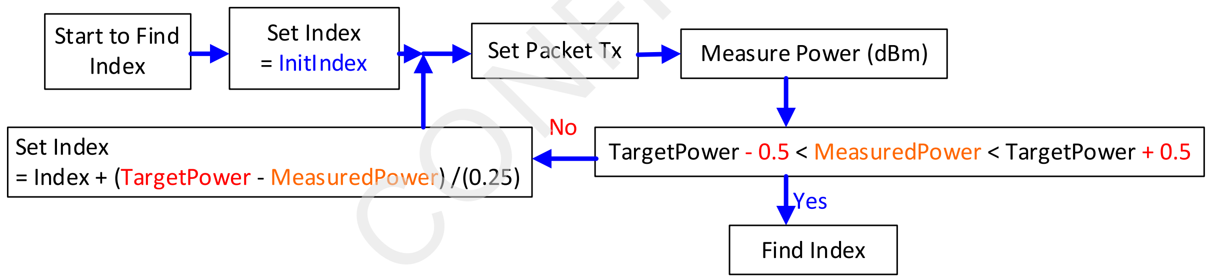


Figure 7: Finding index flow

3.1.3.4 Calibration command and sample

The following command take “ Channel 4, HT40, HTMCS7” as example.
Step 1 : MP start and select channel, bandwith, rate.

```
iwpriv mp_start
iwpriv mp_channel 4
iwpriv mp_bandwidth 40M=1,shortGI=0 // set bandwidth 40M, short GI off
iwpriv mp_rate 135
//rate: sets the rate of Tx packets (i.e.,2 for 1M, 4 for 2M, 11 for 5.5M, 22 for 11M …,
108 for 54M, 128 for MCS0, 129 for MCS1,…, 135 for MCS7)
iwpriv mp_txpower patha=84     // set Tx power index 84
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 and follow figure 6 to find power index.

```
iwpriv mp_txpower patha=80   //turn power index to fit target power
```

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

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

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

While finding Index_MCS7_B40, the Init Index is defined by user or programmer and target power is MCS7-B40 target power level defined before.

After finding all Index_MCS7_B40, use these values to get all Tx gain index in each channel set by interpolation. As 2G channels for example, if the measured index in CH4 is 41 and the measured index in CH10 is 43, all 2G group MCS7-B40 index is shown as below:

Group 1	Group 2	Group 3	Group 4	Group 5
40 (Calculated by Interpolation)	41 (Measured)	42 (Calculated by Interpolation)	43 (Measured)	44 (Calculated by Interpolation)

Table 3-5: The example of finding MCS7 index in 2G band by interpolation

Similarly, all group CCK index is shown as below:

Group 1	Group 2	Group 3	Group 4	Group 5	Group 6
44 (Calculated by Interpolation)	45 (Measured)	46 (Calculated by Interpolation)	47 (Measured)	48 (Calculated by Interpolation)	49 (Measured)

Table 3-6: The example of finding index in 2G band by interpolation

3.1.3.5 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
```

```
#
iwpriv mp_ther
Private Message: 106
[MEM] After do cmd, available heap 12785856
```

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

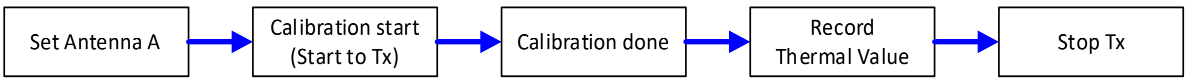


Figure 8: Record thermal value flow

3.2 WiFi performance verification

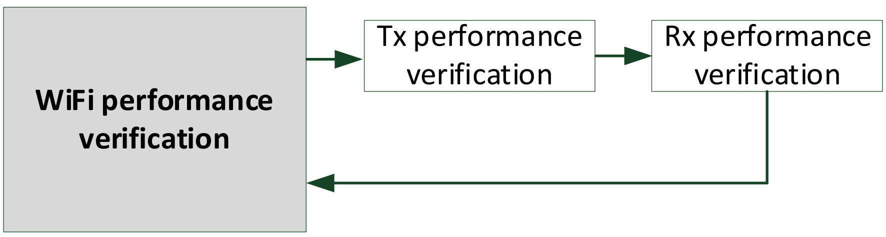


Figure 9: WiFi performance verification flow

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 power index 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	
HTMCS7-B40	**1-Antenna DUT: Main 2-Antenna DUT: Aux**	**CH6**	**Power**	3.3V	Typical: 17dBm, Acceptable Range: +1/-1.5dB
				1.8V	Typical: 11dBm, Acceptable Range: +1/-1.5dB
			EVM	< -28dB	
			Freq. Err.	±15ppm	
			Leakage	< -20dB total	
			Mask	IEEE SPEC. defined	
HTMCS7-B20	**Main**	**CH1**	**Power**	3.3V	Typical: 17dBm, Acceptable Range: +1/-1.5dB
				1.8V	Typical: 11dBm, Acceptable Range: +1/-1.5dB
			EVM	< -28dB	
			Freq. Err.	±15ppm	
			Leakage	< -20dB total	
			Mask	IEEE SPEC. defined	
OFDM 54M	**Main**	**CH7**	**Power**	3.3V	Typical: 18dBm, Acceptable Range: +1/-1.5dB
				1.8V	Typical: 12dBm, Acceptable Range: +1/-1.5dB
			EVM	< -25dB	
			Freq. Err.	±15ppm	
			Leakage	< -15dB total	
			Mask	IEEE SPEC. defined	
CCK 11M	**Main**	**CH7**	**Power**	3.3V	Typical: 20dBm, Acceptable Range: +1/-1.5dB
				1.8V	Typical: 15dBm, Acceptable Range: +1/-1.5dB
			EVM	< 8%	
			Freq. Err.	±15ppm	
			Mask	IEEE SPEC. defined	

Data Rate	Antenna	Channel	Item	Criterion	
HTMCS7-B40	**1-Antenna DUT: Main 2-Antenna DUT: Aux**	**CH38 CH54 CH102 CH151**	**Power**	3.3V	Typical: 15dBm, Acceptable Range: +1/-1.5dB
				1.8V	Typical: 8dBm, Acceptable Range: +1/-1.5dB
			EVM	< -28dB	
			Freq. Err.	±15ppm	
			Leakage	< -20dB total	
			Mask	IEEE SPEC. defined	
MCS7-B20	**Main**	**CH48 CH64 CH144 CH177**	**Power**	3.3V	Typical: 15dBm, Acceptable Range: +1/-1.5dB
				1.8V	Typical: 8dBm, Acceptable Range: +1/-1.5dB
			EVM	< -28dB	
			Freq. Err.	±15ppm	
			Leakage	< -20dB total	
			Mask	IEEE SPEC. defined	
OFDM 54M	**Main**	**CH44 CH60 CH120 CH165**	**Power**	3.3V	Typical: 16dBm, Acceptable Range: +1/-1.5dB
				1.8V	Typical: 9dBm, Acceptable Range: +1/-1.5dB
			EVM	< -25dB	
			Freq. Err.	±10ppm	
			Leakage	< -15dB total	
			Mask	IEEE SPEC. defined	

Table 3-7: 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

Please refer to Section 3.1.3.4, you can calculate the following index information:

Index_MCS7_B20_Group1 = Index_MCS7_B40_Group1 + (target power diff of MCS7-B20 to MCS7-B40)*4
Index_MCS7_B20_Group2 = Index_MCS7_B40_Group2 + (target power diff of MCS7-B20 to MCS7-B40)*4
Index_MCS7_B20_Group3 = Index_MCS7_B40_Group3 + (target power diff of MCS7-B20 to MCS7-B40)*4
Index_MCS7_B20_Group4 = Index_MCS7_B40_Group4 + (target power diff of MCS7-B20 to MCS7-B40)*4
Index_MCS7_B20_Group5 = Index_MCS7_B40_Group5 + (target power diff of MCS7-B20 to MCS7-B40)*4

Index_OFDM_Group1= Index_MCS7_B40_Group1 + (target power diff of 54M-1T to MCS7-B40)*4
Index_OFDM_Group2= Index_MCS7_B40_Group2 + (target power diff of 54M-1T to MCS7-B40)*4
Index_OFDM_Group3= Index_MCS7_B40_Group3 + (target power diff of 54M-1T to MCS7-B40)*4
Index_OFDM_Group4= Index_MCS7_B40_Group4 + (target power diff of 54M-1T to MCS7-B40)*4
Index_OFDM_Group5= Index_MCS7_B40_Group5 + (target power diff of 54M-1T to MCS7-B40)*4

For example:
You want to verify WiFi Tx Performance and you can follow step as below.

Step 1: Initial WiFi Step
Please refer to Section 3.1.1

Step 2: start Tx
Ex: Rate MCS7, Bandwidth40, Channel 6, Tx_target = 17dBm, **Index_MCS7_B40_Group3** = 86 (the value is from 3.1.3.4 record)

```
iwpriv mp_channel 4
iwpriv mp_bandwidth 40M=1,shortGI=0
iwpriv mp_txpower patha=86
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)
iwpriv mp_hwtx period=100,len=1500,count=0
```

Step 3: Measure Tx performance and check the performance with criterion table.
CH6 HT40 MCS7 criterion :

Data Rate	Channel	Item	Criterion	
HTMCS7-B40	**CH6**	**Power**	3.3V	Typical: 17dBm, Acceptable Range: +1/-1.5dB
			1.8V	Typical: 11dBm, Acceptable Range: +1/-1.5dB
		EVM	< -28dB	
		Freq. Err.	±15ppm	
		Leakage	< -20dB total	
		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 3-6. And the command is list in [3.1.1.2](#).

3.2.2 Rx performance verification

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
MCS7-B40	1-Antenna DUT: Main 2-Antenna DUT: Aux	CH6	Sensitivity	< -64dBm
MCS7-B20	Main	CH1		< -67dBm
OFDM 54M	Main	CH7		< -71dBm
CCK 11M	Main	CH13		< -82dBm

5Ghz				
Data Rate	Antenna	Channel	Item	Criterion
MCS7-B40	1-Antenna DUT: Main 2-Antenna DUT: Aux	CH118	Sensitivity	< -64dBm
MCS7-B20	Main	CH44		< -67dBm
OFDM 54M	Main	CH100		< -71dBm

Table 3-8: 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.

For example:
You want to verify WiFi Rx Performance and you can follow step as below.

Step 1: Initial WiFi Step
Please refer to Section 3.1.1.

Step 2: Start Rx Test(use 802.11b, channel 1, 11Mbps, Bandwidth 20)

```
iwpriv mp_channel 1
iwpriv mp_bandwidth 40M=0,shortGI=0
iwpriv mp_arx start
iwpriv mp_reset_stats
```

Step 3: Get report

```
iwpriv mp_arx phy
```

Step 4: Stop Rx Test

```
iwpriv mp_arx stop
```

NOTE

According to rfe_type, Aux-antenna Rx performance verification is needed if it exists. The recommended test items are listed at table 3-8. And the command is list in 3.1.1.2.

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.

No	Steps	Commands/Descriptions	Band rate	Remark
1	MP start	iwpriv mp_start	1500000	
2	WiFi Calibration & Verification	iwpriv commands	1500000	
3	Switch to Bluetooth	ATM2=bt_power,on ATM2=gnt_bt,bt ATM2=bridge	1500000	Please refer to 5.1.1.
4	Open BT Uart interface	Refer to the sample codes in Realtek Bluetooth MP Kit Setup Package	115200	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
5	BT Tx calibration (gain k, flatness k)			
6	Read BT thermal meter value			
7	BT TRx verification			
8	Close BT Uart interface			
9	Switch to Wifi	ATM2=bridge,close	115200	Please refer to 5.1.1.
		ATM2=gnt_bt,wifi	1500000	
10	Write OTP	iwpriv commands	1500000	Please refer to 12. Appendix for Calibration Data Contents Spec * Write WiFi calibration data+MAC address * Write BT calibration data+MAC address.
11	MP stop	iwpriv mp_stop	1500000	

Table 4-1: WiFi & BT Mass Production flow

5 BT Mass Production Flow

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

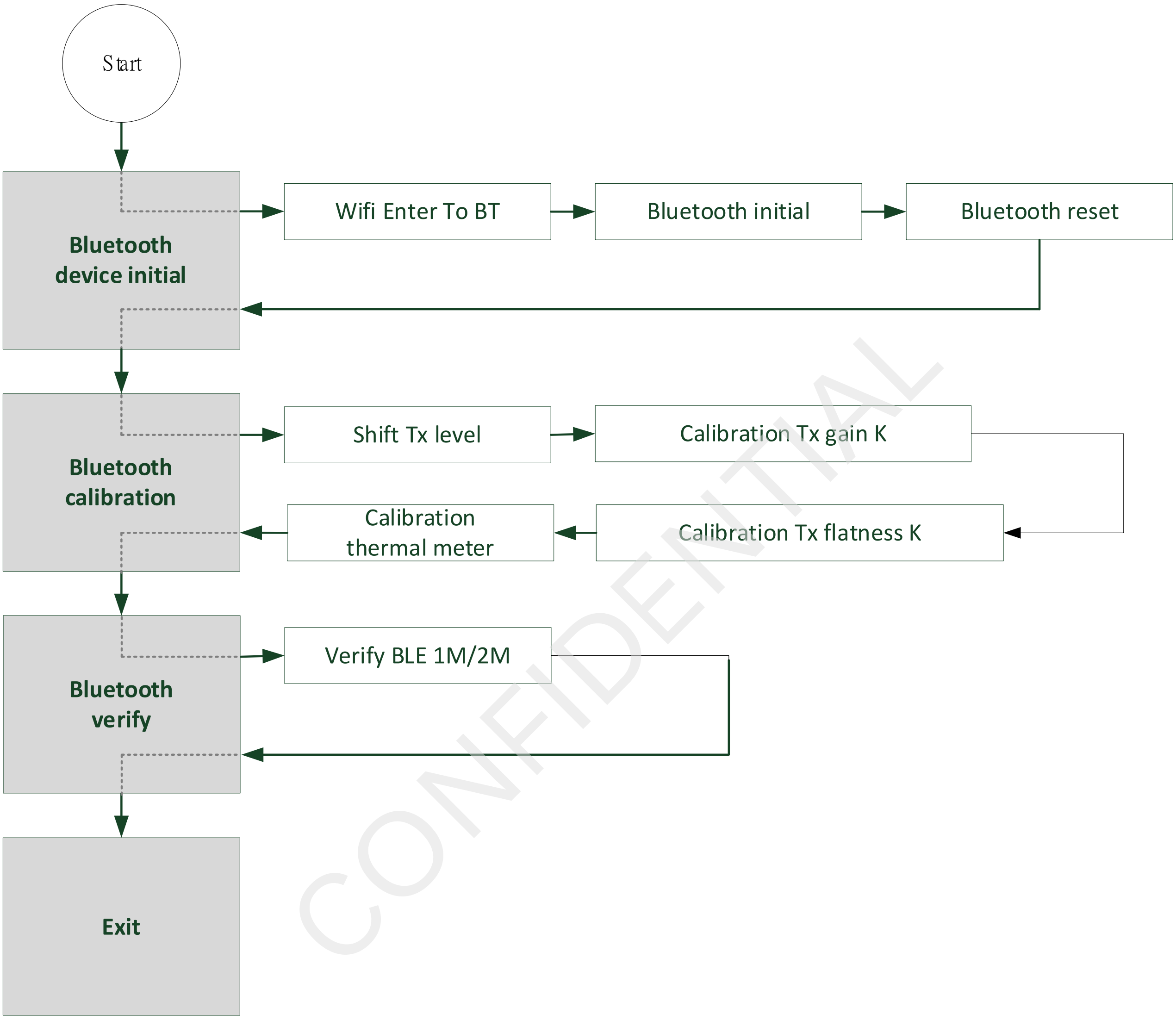


Figure 10 BT Mass Production Test flow

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 compete the BT initial stage, the Bluetooth device must set to factory default.

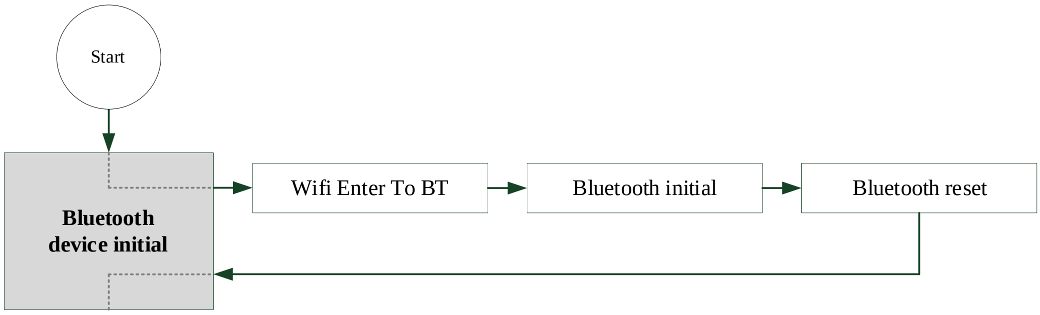


Figure 11: Bluetooth Device Initial flow

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".

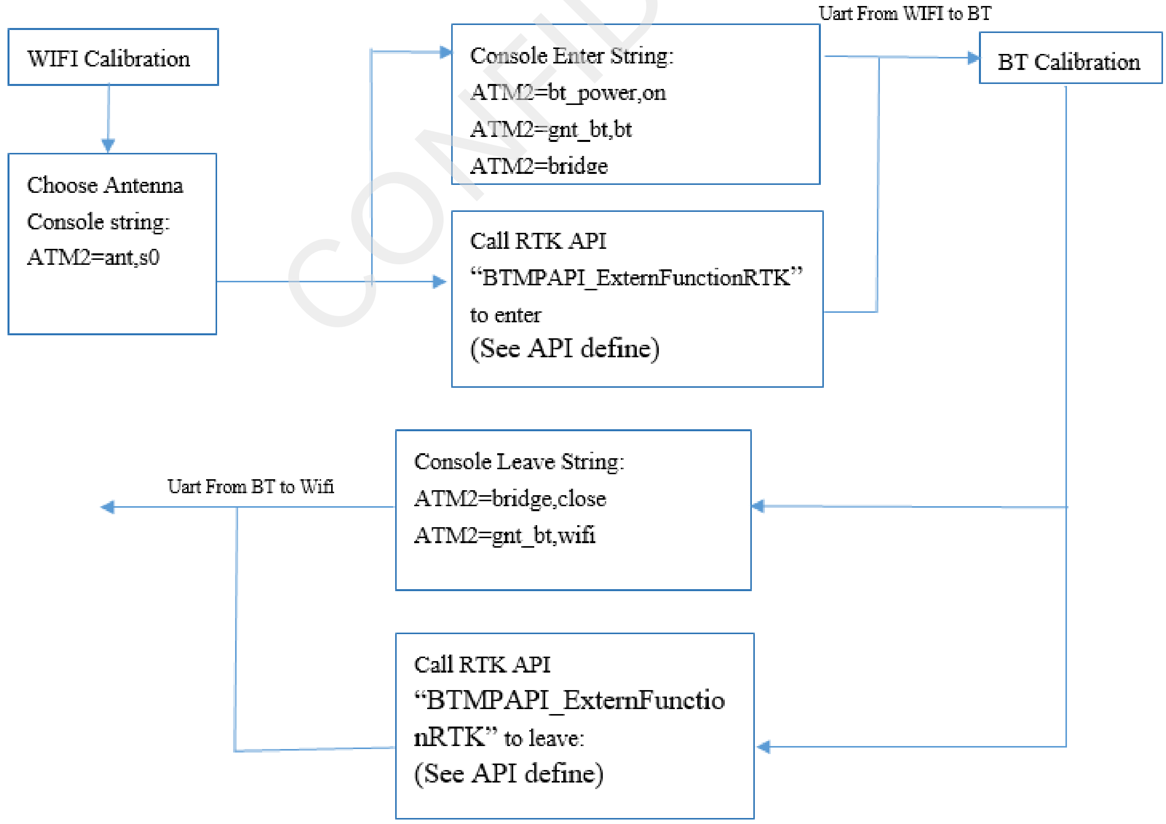


Figure 12: Wifi bridge to BT

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. The program sample code by dynamic is as follow:

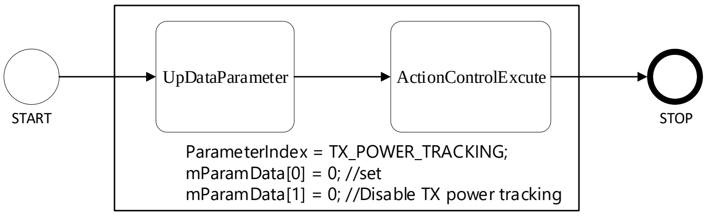


Figure 13: Disable TX power tracking flow

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”.

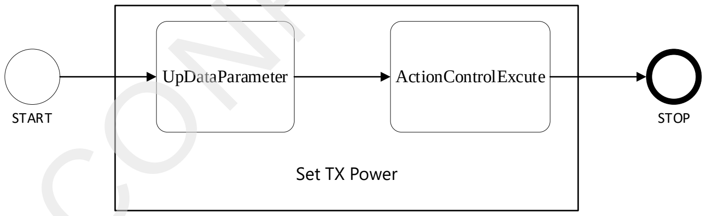


Figure 14: Set TX Power Level To RAM flow

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. If only BLE is supported, the API format will not change, but the settings of BR 1M/EDR2M/EDR3M will be ignored.

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 code by RF dynamic link library is as follows:

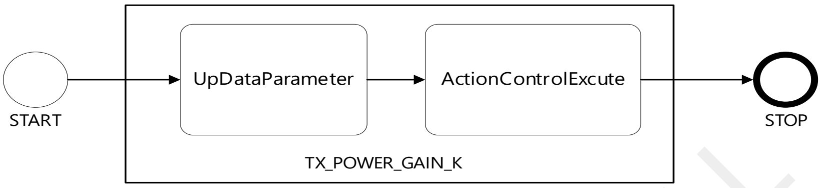


Figure 15: Set TX gain cal (K) value

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 5-1: Set flatness Value command table

The flatness settings of APIs are defined below:
A simple example code by RF dynamic link library is as follows:

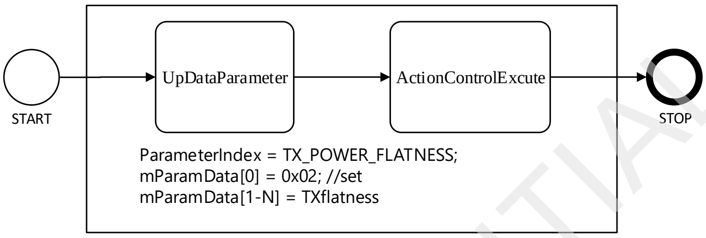


Figure 16: Set flatness Value flow

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 5-2: Set TX path loss module Value Table

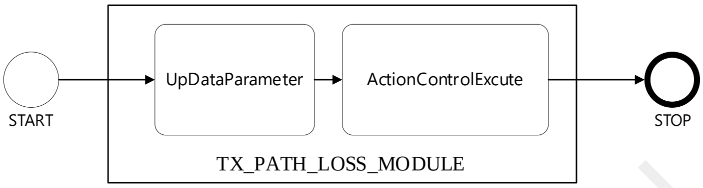


Figure 17: Set TX Path Loss

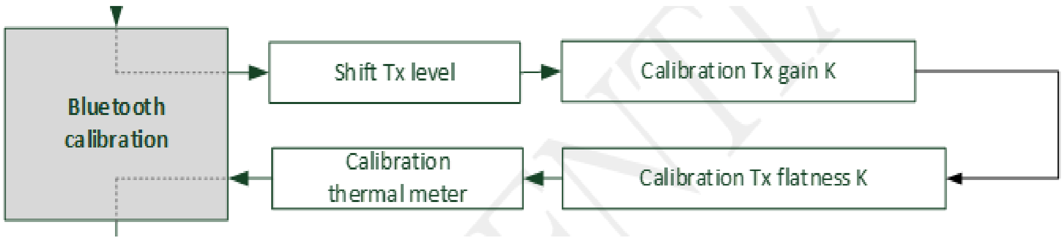
Sample:

```
pParam->ParameterIndex = TX_PATH_LOSS_MODULE;
pParam->mParamData[0] = 0x03; // set value
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.

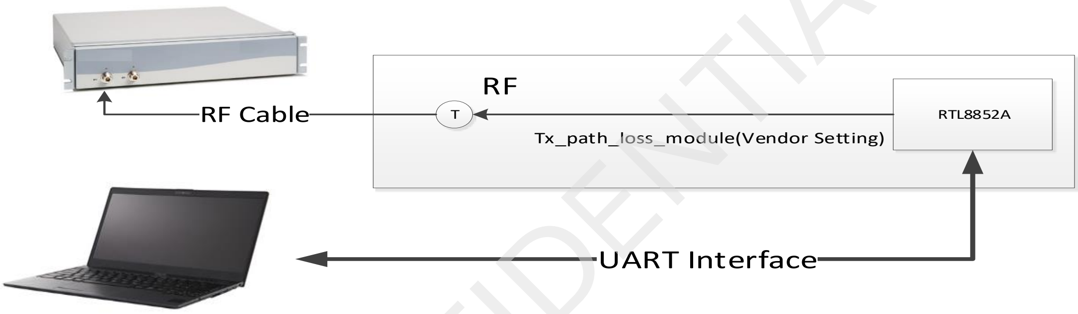


Figure 18: Bluetooth calibration Tx Power schematic

5.3.1 Set Power TX Level to RAM

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

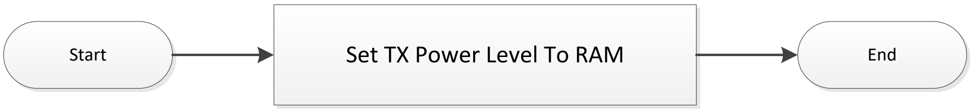


Figure 19: Set Power TX Level to RAM Flow

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.

5.3.2 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.**

In this trim TX power flow, use BLE 1M Packet type and channel 19 (2440MHz) for calibration. The calibration flow is as follows:

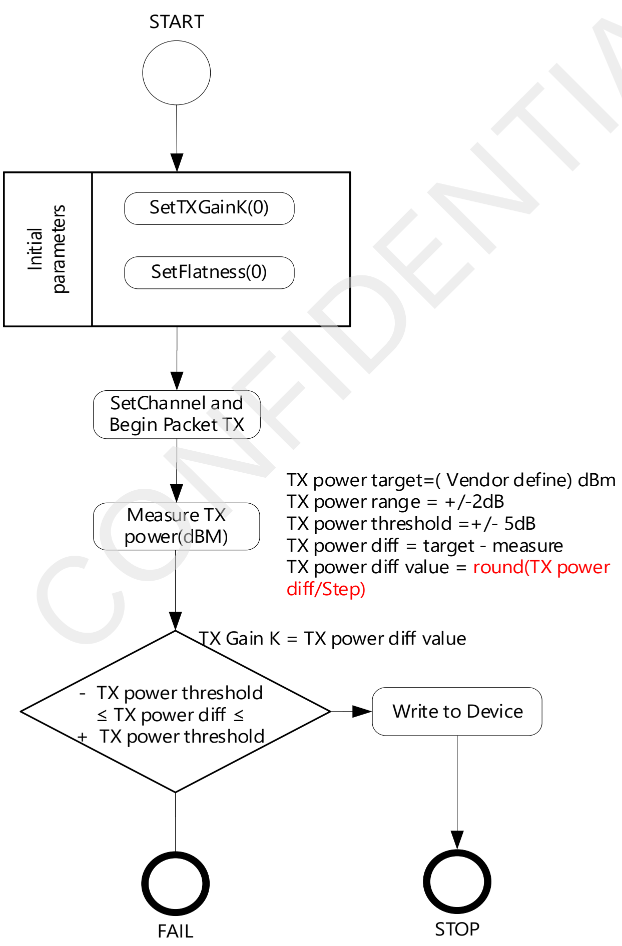


Figure 20 Calibration to Target Power

Ex. Sample Code by C++:
(1) Initial

```
//Set TX gain cal (K) to 0
pParam->ParameterIndex = TX_POWER_GAIN_K;
pParam->mParamData[0] = 0x01; //set
pParam->mParamData[1] = 0;
pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
pBluetoothModule->ActionControlExcute(pBluetoothModule);
//Set TX flatness to 0x00000000
pParam->ParameterIndex = TX_POWER_FLATNESS;
pParam->mParamData[0] = 0x02; //set
pParam->mParamData[1] = 0x00;
pParam->mParamData[2] = 0x00;
pParam->mParamData[3] = 0x00;
pParam->mParamData[4] = 0x00;
pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
pBluetoothModule->ActionControlExcute(pBluetoothModule);
```

(2)Start Channel 19 BLE 1M TX

```
pParam ->ParameterIndex =LE_TX_DUT_TEST_CMD
pParam-->mChannelNumber=0x13;
pParam->mPayloadType=0x00;   //Payload type = PRBS9
pParam->mTxGainIndex=0x00; //use FW Keep Tx gain
pParam->PHY = 1; // 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);
```

(3) Measured TX by Bluetooth test instrument(ex. Letepoint IQNxN)
(4)Calculate tx Gain K Value
(5)Stop BLE TX

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

(6)Set new Tx gain k value to RAM

```
pParam->ParameterIndex = TX_POWER_GAIN_K;
pParam->mParamData[0] = 0x01; //set
pParam->mParamData[1] = New Tx gain K;
pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
pBluetoothModule->ActionControlExcute(pBluetoothModule);
```

(7)Write to Device.

5.3.3 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.**

In this trim flatness TX power flow, use BLE 1M Packet type and channel 4(2410MHz), channel 15(2432MHz),channel 25(2452MHz) and channel 35(2472MHz) for calibration. The calibration flow is as follows:

Measure Number	1st	2nd	3rd	4th
Measure channel	**4**	**15**	**25**	**35**

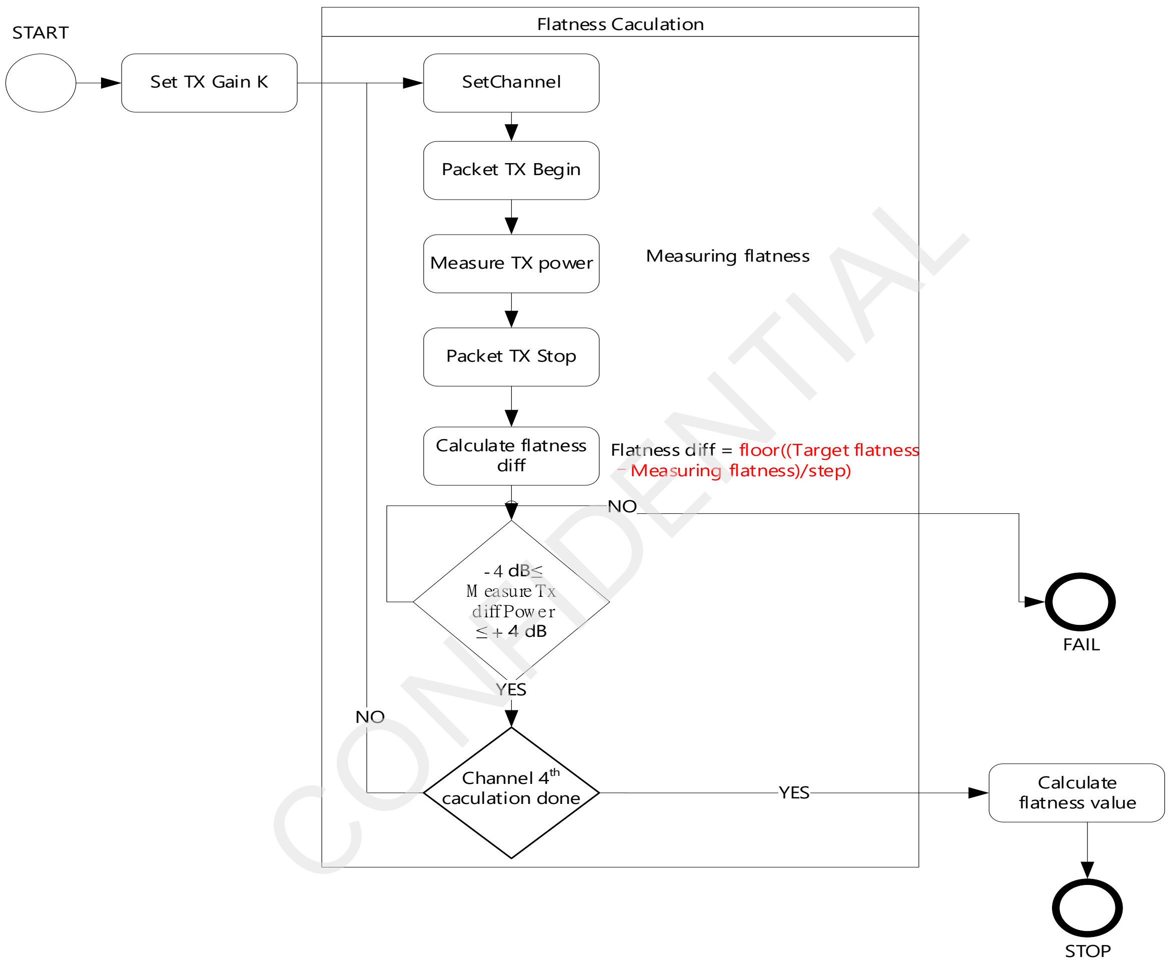


Figure 21: Bluetooth TX Power Flatness Calibration flow

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 TX power. 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, ch0 ~ 10	MidLow Channel Flatness Ch11~ch21	MidHigh Channel Flatness Ch22 ~ 30	High Channel Flatness Ch31 ~ 39

Table 5-3: 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)
4	-0.6	floor((0-(-0.6))/0.25)	2	0x02 -> 0x08	0xfbfd0102	0xecf40408
15	-0.3	floor((0-(-0.3))/0.25)	1	0x01-> 0x04		
25	0.7	floor((0-(0.7))/0.25)	-3	0Xfd-> 0Xf4		
35	1.2	floor((0-(1.2))/0.25)	-5	0xfb-> 0xec		

Table 5-4: The TX power flatness value define sample table (4byte)

Ex. Sample Code by C++:
(1)First to set Tx gain K
(2)Set TX flatness to 0x00000000

```
pParam->ParameterIndex = TX_POWER_FLATNESS;
pParam->mParamData[0] = 0x02; //set
pParam->mParamData[1] = 0x00;
pParam->mParamData[2] = 0x00;
pParam->mParamData[3] = 0x00;
pParam->mParamData[4] = 0x00;
pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
pBluetoothModule->ActionControlExcute(pBluetoothModule);
```

(3)Start Channel 04/15/25/35 BLE 1M TX

```
pParam ->ParameterIndex =LE_TX_DUT_TEST_CMD
pParam-->mChannelNumber=0x4; //Chanel 0x04/0x0F/0x19/0x23
pParam->mPayloadType=0x00;    //Payload type = PRBS9
pParam->mTxGainIndex=0x00; //use FW Keep Tx gain
pParam->PHY = 1; // 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);
```

(4) Measured TX by Bluetooth test instrument(ex. Letepoint IQNxN)
(5) Calculate TX Gain Flatness[0~3] Value
(6) Stop BLE TX

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

(7) Repeat to step 2 and change channel
(8)Set new Tx flatnees k value to RAM

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

(9)Write to Device.

5.3.4 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 code show as

```
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

5.4.1 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:

Verify BLE Tx	Test Item	Sub Test Item	Payload Type	Channel	Criterion
					Bluetooth Spec: Power domain = 1.8V Target tx power = 7 dBm Power domain = 3.3V Target tx power = 8 dBm
	BLE Output Power	Average Power	PRBS9	CH0-2402MHz	**Power domain = 1.8V** Settable range:-12.5dBm ~ 7.5 dBm **Power domain = 3.3V** Settable range:-12.5dBm ~ 12.5 dBm NOTE: Max **variation** = target power value +/- 2 dB*
				CH19-2440MHz	
				CH39-2480MHz	
	Modulation Characteristics	Delta F1 Avg.	BT_PAYLOAD_TYPE_1111_0000	CH19-2440MHz	225 kHz ~ 275 kHz
		Delta F2 Max.	BT_PAYLOAD_TYPE_1010		≥ 185 kHz
		Modulation Index	None		≥ 0.8

Table 5-5: Bluetooth BLE TX criterion(BT4.0)

For example, the step by step:

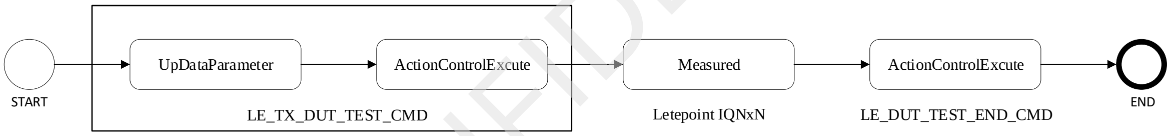


Figure 22: Verify Bluetooth LE TX Performance flow (BT4.0)

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	Channel = 0/19/39

BLE	Average Power	pParam ->ParameterIndex =LE_TX_DUT_TEST_CMD
		pParam-->mChannelNumber=0x00; //(change channel :0x13,0x27)
		pParam->mPayloadType=0x00; //Payload type = PRBS9
		pParam->mTxGainIndex=0x00; //use FW Keep Tx gain
		pParam->PHY = 1; // 1:LE 1M 2:LE 2M 3: LE Coded PHY
		pParam->ModulationIndex = 1; // 0: standard modulation 1: stable modulation

BLE	Delta F1	pParam ->ParameterIndex =LE_TX_DUT_TEST_CMD
		pParam-->mChannelNumber=0x00; //(change channel :0x13,0x27)
		pParam->mPayloadType=0x01; //Payload type = 11110000
		pParam->mTxGainIndex=0x00; //use FW Keep Tx gain
		pParam->PHY = 1; // 1:LE 1M 2:LE 2M 3: LE Coded PHY
		pParam->ModulationIndex = 1; // 0: standard modulation 1: stable modulation

BLE	Delta F2	pParam ->ParameterIndex =LE_TX_DUT_TEST_CMD
		pParam-->mChannelNumber=0x00; //(change channel :0x13,0x27)
		pParam->mPayloadType=0x02; //Payload type = 10101010
		pParam->mTxGainIndex=0x00; //use FW Keep Tx gain
		pParam->PHY = 1; // 1:LE 1M 2:LE 2M 3: LE Coded PHY
		pParam->ModulationIndex = 1; // 0: standard modulation 1: stable modulation

```
pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
```

(3)Execute active job - Run LE DUT TX

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

(4) Measured by Bluetooth test instrument(ex. Letepoint IQNxN)

(5)Stop BLE TX

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

5.4.2 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: Power domain = 1.8V Target tx power = 7 dBm Power domain = 3.3V Target tx power = 8 dBm
2M LR S2 LR S8	BLE Output Power	Average Power	PRBS9	CH01-2404MHZ CH19-2440MHZ CH38-2478MHZ	**Power domain = 1.8V** Settable range:-12.5dBm ~ 7.5 dBm **Power domain = 3.3V** Settable range:-12.5dBm ~ 12.5 dBm NOTE: Max **variation** = target power value +/- 2 dB*
2M	Carrie freq. offset & drift	None	PRBS9		2M: < 20 kHz
LR S2			PRBS9		S8: < 19.2 kHz
LR S8			PRBS9		
2M	Modulation Characteristics	Delta F1 Avg.	BT_PAYLOAD_TYPE_1111_0000		2M: 450 ~ 550 kHz S8: 225 ~ 275 kHz
LR S2		Delta F2 Max.	BT_PAYLOAD_TYPE_1010		≥ 370 kHz
LR S8		Modulation Index	None		≥ 0.8

Table 5-6: Bluetooth BLE 5.0 TX criterion

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

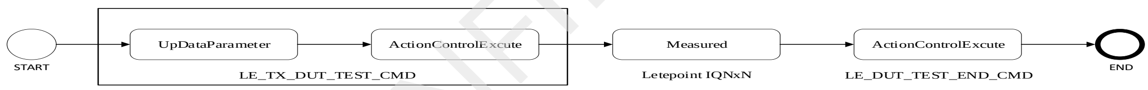


Figure 23: Verify Bluetooth LE TX Performance flow(BT5.0)

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

Test Item	Channel = 1/19/38

Average Power	pParam ->ParameterIndex =LE_TX_DUT_TEST_CMD
	pParam -> mChannelNumber=0x01;// change channel :0x13,0x26
	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 :0x13,0x26
	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 :0x13,0x26
	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)Stop BLE TX

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

5.4.3 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
Low (CH0-2402MHZ)	PRBS9	PER<= 30.800 %	-85dBm
Middle (CH19-2440MHZ)			
High (CH39-2480MHZ)			

Table 5-7: 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 :

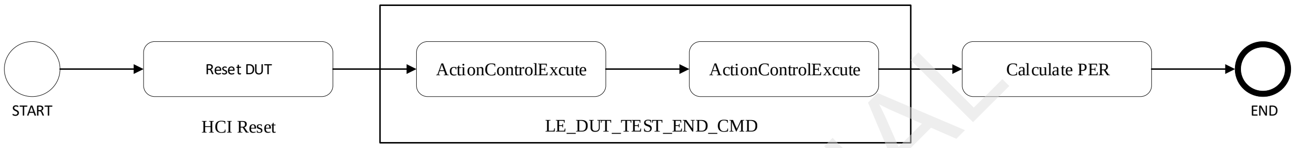


Figure 24: The recommended test items of Bluetooth BLE Rx flow

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

Item	Payload type	code

Sensitivity	PRBS9	pParam ->ParameterIndex =LE_RX_DUT_TEST_CMD
		pParam ->mChannelNumber=0x00;// //change channel :0x13,0x27
		pParam ->mPayloadType=0x02; //Payload type =10101010
		pParam->PHY = 1 ; // 1:LE 1M 2:LE 2M 3: LE Coded PHY
		pParam->ModulationIndex = 1; // 0: standard modulation 1: stable modulation

pBluetoothModule->UpDataParameter(pBluetoothModule,pParam);
(3)To setting Parameter with the Bluetooth test instrument.
(4)Run LE Packet Rx
pBluetoothModule->ActionControlExcute(pBluetoothModule);
(5)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
pBluetoothModule->ActionReport(pBluetoothModule, REPORT_LE_RX, &mBT_DEVICE_REPORT);
```

(6) Calculate PER
If (PER< limit), go to step (3), else Stop

5.4.4 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
01/19/38	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 5-8: 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 :

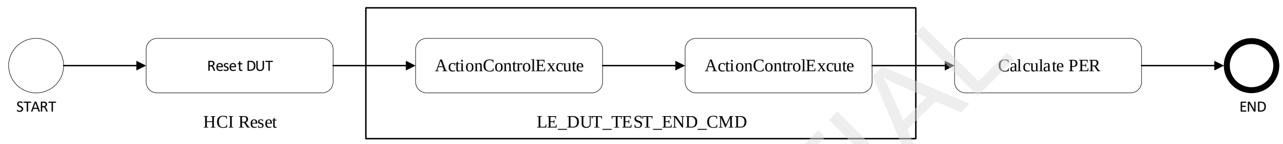


Figure 25: Verify Bluetooth BLE5.0 Rx

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

Item	**Packet type**	

Sensitivity	LE2M	mChannelNumber=0x01; (change channel :0x13,0x26)
		mPayloadType=0x01; //PRBS9
		pParam->PHY = 2 ; // 1:LE 1M 2:LE 2M 3: LE Coded PHY
		pParam->ModulationIndex = 1

Sensitivity	LR	mChannelNumber=0x01; (change channel :0x13,0x26)
		mPayloadType=0x01; //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
pBluetoothModule->ActionReport(pBluetoothModule, REPORT_LE_RX, &mBT_DEVICE_REPORT);
```

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

6 Write Data to Storage and Check Contents

Write board-dependent information into respective calibration data offset, this information include MAC address, calibrated power index, 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 ' RTL8721DA_RTL8721DC_RTL8721DG_default_OTP.map' file).

iwpriv config_set wmap,	0x020,	50	50	50	50	50	50	50	50	50	50	50	02	FF	FF	FF	FF
iwpriv config_set wmap,	0x030,	FF	FF	50	50	50	50	50	50	50	50	50	50	50	50	50	50
iwpriv config_set wmap,	0x040,	02	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
iwpriv config_set wmap,	0x0C0,	FF	FF	FF	FF	FF	FF	FF	FF	FF	48	66	FF	FF	FF	FF	FF
iwpriv config_set wmap,	0x110,	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	00	E0	4C	00	00	01
iwpriv config_set wmap,	0x130,	FF	21	FF	11	09	FF	FF	FF	00	11	00	30	30	FF	FF	FF

0x20~0x25	2.4G CCK power index
0x26~0x2A	2.4G BW40 power index
0x2B	2.4G Difference
0x32~0x3F	5G BW40 power index
0x40	5G Difference
0xC9	Crystal Calibration
0xCA	Thermal meter
0x11A~0x11F	WiFi MAC Address
0x131-0x13A	Default values should not be changed， but need to be writen to efuse
0x13B~0x13C	Country code
0x13D	DC power supply mode 0x13D=0xFF or 0x00 for 3.3V 0x13D=0x01 for 1.8V

Table 6-1: 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 ' RTL8721DA_RTL8721DC_RTL8721DG_default_OTP.map ' file).
The OTP define show as:

iwpriv config_set wmap,	0x1B0,	FF	FF	FF	FF	00	E0	4C	80	00	01	FF	FF	FF	FF	10	FF
iwpriv config_set wmap,	0x1C0,	FF	FF	FF	FF	3F	07	FF	3B	3B	FF	FF	3B	FF	FF	FF	FF
iwpriv config_set wmap,	0x1D0,	66	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF

0x1B4 ~ 0x1B9	BD_ADDRESS
0x1BE[0]	rsvd0 (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)[3]
0x1BE[7:6]	reserved (default 0)
0x1BF	TX Gain K
0x1C0 ~ 0x1C3	Flatness K
0x1C4	iqm_txgain_10dBm_raw_index 3.3V: 3Fh , 1.8V:45h
0x1C5	lbt_ant_gain 3.3V: 07h , 1.8V: 0h
0x1C7	iqm_max_txgain_LE1M 3.3V:3Bh , 1.8V:44h
0x1C8	iqm_max_txgain_LE2M 3.3V:3Bh , 1.8V:44h
0x1CB	iqm_max_txgain_LR 3.3V:3Bh , 1.8V:44h
0x1D0	tmeterx4_txgaink_module(default 0x66)

Table 6-2: 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.
[3]. If iqm tx gain 10db raw index are been done please enable the lbt 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:

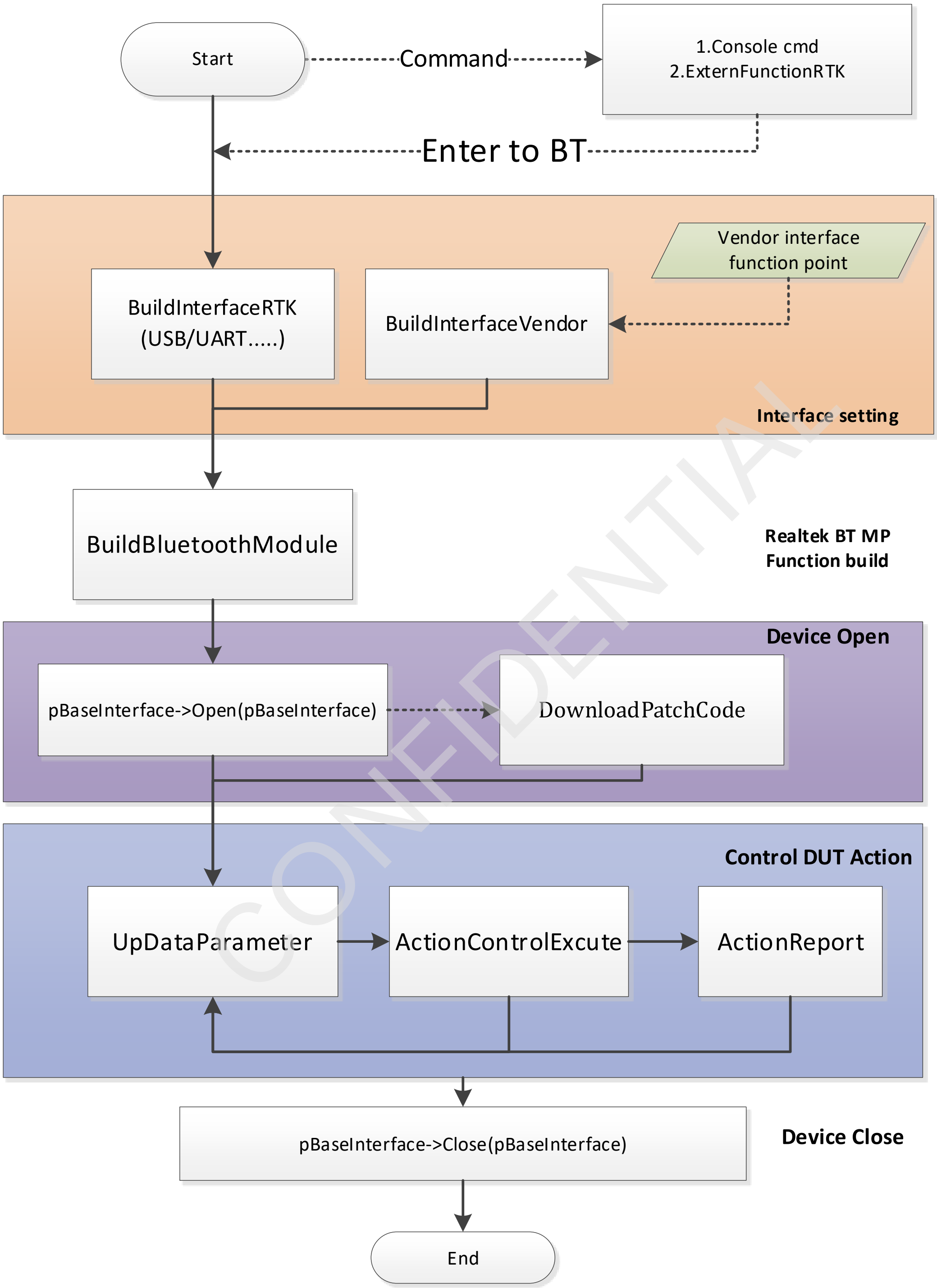


Figure 26: Mass Production API use flow

7.1 Use the API to enter BT HCI Mode: BTMPAPI_ExternFunctionRTK

BTMPAPI_ExternFunctionRTK	
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	

<pre>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);</pre>	

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	

Table 7-1: BTMPAPI_ ExternFunctionRTK API

7.2 Interface Setting

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.

7.2.1 BTMPAPI_BuildInterfaceVendor: Use the customer's interface

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:

```
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 7-2: BTMPAPI_BuildInterfaceVendor API

7.2.2 BTMPAPI_BuildInterfaceRTK: Use the RTK's interface

BTMPAPI_ BuildInterfaceRTK	
This function uses Realtek default functions to open, close, send, receive and wait functions.	
Syntax	

<pre>int BTMPAPI_BuildInterfaceRTK(
    BASE_INTERFACE_MODULE **ppBaseInterface,
    BASE_INTERFACE_MODULE *pBaseInterfaceModuleMemory,
    unsigned int InterfaceType,
    unsigned char   PortNo,
    unsigned long Baudrate
    );</pre>	

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	

Table 7-3: BTMPAPI_ BuildInterfaceRTK API

7.3 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 7-4: BTMPAPI_ BuildInterfaceRTK API

7.4 Device Open

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

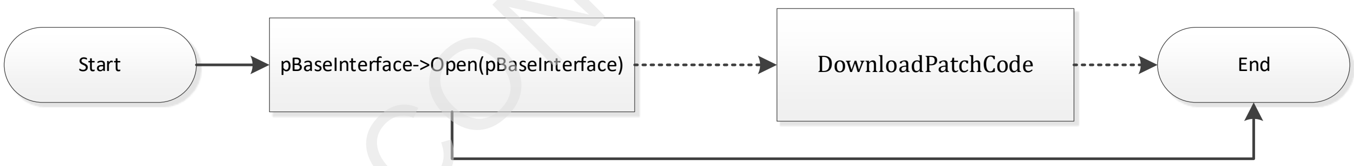


Figure 27: Device Open flow

7.4.1 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);
```

7.4.2 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 7-5: DownloadPatchCode API

Use the API to enter BT HCI Use the API to enter BT HCI The Control function includes three main functions that are "UpDataParameter", ActionControlExcute" and ActionReport".

7.4.3 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 7-6: 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.

7.4.4 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 7-7: ActionControlExcute API

According to “ParameterIndex” in the “BT_PARAMETER” structure, “ActionControlExcute” performs different functions, such as “PACKET_TX_START”, “PACKET_TX_STOP”, etc.

7.4.5 ActionReport

ActionReport	
Report to the execution result information.	
Syntax	

<pre>typedef int
(*BT_DLL_MODULE_FP_ACTION_REPORT)(
    BASE_BTMPDLL_MODULE *pDLLBtBaseModule,
    int ActiceItem,
    BT_DEVICE_REPORT *pReport
    );</pre>	

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 7-8: 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”.

7.5 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 BLE Direct Test Mode (BQB Test)

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

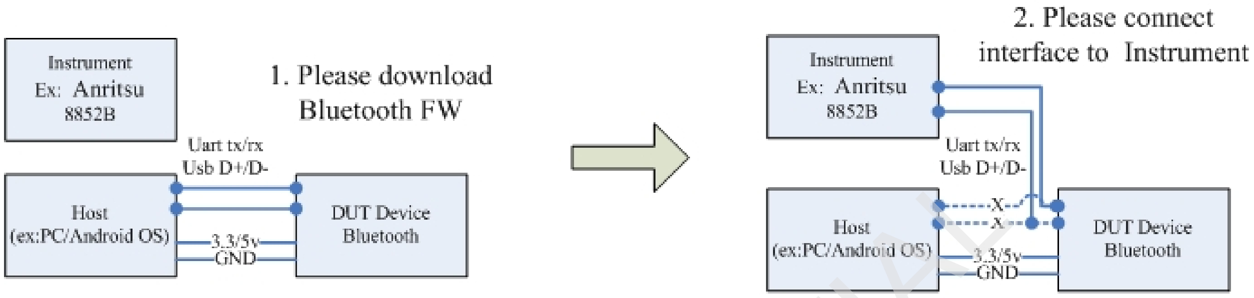


Figure 28: BLE Direct Test Mode

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:

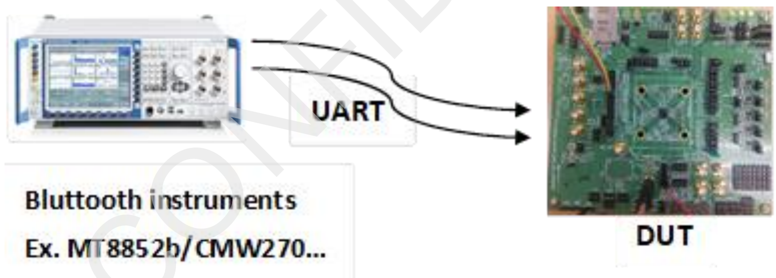


Figure 29: BLE Direct Test Mode connecter

8.1.2 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 8-1 : 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 9-1: 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 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 9-2:BLE TX PHY index

9.3 BLE_MODULATION_TYPE

NAME	INDEX	
STANDARD_MODULATION_INDEX	0	
STABLE_MODULATION_INDEX	1	

Table 9-3: BLE_MODULATION_TYPE

9.4 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 9-4: PAYLOAD_TYPE index for BLE

9.5 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 9-5: PACKET_HEADER

9.6 ActionControl Index

Command	Index
HCI_RESET	0
TEST_MODE_ENABLE	1
SET_XTAL	10
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_READ_TX_POWER_INFO	38
SET_ANT_INFO	40
SET_ANT_DIFF_S0S1	41
TX_POWER_TRACKING	42
SET_K_TX_CH_PWR	43
TX_POWER_GAIN_K	45
TX_POWER_FLATNESS	46
TX_PATH_LOSS_MODULE	47
SetTxPower_8822C	51

Table 9-6 ActionControl Index

9.7 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 9-7: 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 10-1: UpDataParameter define: BT_PARAM

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 10-2: 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 10-3: BT BLE5.0_Rx_PKT_TYPE

BLE5.0_MODULATION_TYPE

NAME	INDEX
STANDARD_MODULATION_INDEX	0
STABLE_MODULATION_INDEX	1

Table 10-4: BT BLE5.0_MODULATION_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 10-5: BT LE_PAYLOAD_TYPE

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;

};
```

11 Appendix: BT 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 11-1: 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 11-2: Flatness Define Table

11.3 Tx power Table

Power domain : 1.8V and 3.3V

Tx power Index	GFSK/BLE power (dBm)		Tx power Index	GFSK/BLE power (dBm)	
	Power domain =1.8V	Power domain =3.3V		Power domain =1.8V	Power domain =3.3V
0x12		-12.5	0x2c	-5.0	0.5
0x13		-12.0	0x2d	-4.5	1.0
0x14		-11.5	0x2e	-4.0	1.5
0x15		-11.0	0x2f	-3.5	2.0
0x16		-10.5	0x30	-3.0	2.5
0x17		-10.0	0x31	-2.5	3.0
0x18		-9.5	0x32	-2.0	3.5
0x19		-9.0	0x33	-1.5	4.0
0x1A		-8.5	0x34	-1.0	4.5
0x1B		-8.0	0x35	-0.5	5.0
0x1C		-7.5	0x36	0.0	5.5
0x1D	-12.5	-7.0	0x37	0.5	6.0
0x1E	-12.0	-6.5	0x38	1.0	6.5
0x1F	-11.5	-6.0	0x39	1.5	7.0
0x20	-11.0	-5.5	0x3a	2.0	7.5
0x21	-10.5	-5.0	0x3b	2.5	**8.0**
0x22	-10.0	-4.5	0x3c	3.0	8.5
0x23	-9.5	-4.0	0x3d	3.5	9.0
0x24	-9.0	-3.5	0x3e	4.0	9.5
0x25	-8.5	-3.0	0x3f	4.5	10.0
0x26	-8.0	-2.5	0x40	5.0	10.5
0x27	-7.5	-2.0	0x41	5.5	11.0
0x28	-7.0	-1.5	0x42	6.0	11.5
0x29	-6.5	-1.0	0x43	6.5	12.0
0x2a	-6.0	-0.5	0x44	**7.0**	12.5
0x2b	-5.5	0.0	0x45	7.5	

Table 11-3: Power domain 1.8V and 3.3V Tx power Table

12 Appendix: WLAN and BT Controller Calibration Data Contents Spec

Bytes	Contents	Description	Value
20h	Path A 2.4G CCK-1TX Power Index (Absolute Value)	Path A CCK Power Index for Ch 1,2, Range 0~127, 0.25dB/step.	50 h
21h		Path A CCK Power Index for Ch 3, 4, 5, Range 0~127, 0.25dB/step.	50 h
22h		Path A CCK Power Index for Ch 6, 7 ,8, Range 0~127, 0.25dB/step.	50 h
23h		Path A CCK Power Index for Ch 9, 10, 11, Range 0~127, 0.25dB/step.	50 h
24h		Path A CCK Power Index for Ch 12, 13, Range 0~127, 0.25dB/step.	50 h
25h		Path A CCK Power Index for Ch 14, Range 0~127, 0.25dB/step.	50 h
26h	Path A 2.4G BW40-1S Tx Power Index (Absolute Value)	Path A 2G BW40-1S Power Index for Ch 1, 2, Range 0~127, 0.25dB/step.	50 h
27h		Path A 2G BW40-1S Power Index for Ch 3, 4, 5, Range 0~127, 0.25dB/step.	50 h
28h		Path A 2G BW40-1S Power Index for Ch 6, 7 ,8, Range 0~127, 0.25dB/step.	50 h
29h		Path A 2G BW40-1S Power Index for Ch 9, 10, 11, Range 0~127, 0.25dB/step.	50 h
2Ah		Path A 2G BW40-1S Power Index for Ch 12, 13, 14 Range 0~127, 0.25dB/step.	50 h
2Bh	Path A 2.4G BW20-1S Tx Power Index Difference Path A 2.4G OFDM-1Tx Power Index Difference	Power Index Difference between BW20-1S and BW40-1S. Bit[7:4] : Path A 2G Offset, Range –8~7,0.5dB/step. Power Index Difference between OFDM-1Tx and BW40-1S. Bit[3:0] : Path A 2G Offset, Range –8~7,0.5dB/step.	02 h
2Ch~31h	Reserved	Reserved for Realtek.	-
32h	Path A 5G BW40-1S Tx Power Index (Absolute Value)	Path A 5G BW40-1S Power Index for Ch 36, 38, 40, Range 0~127.	50 h
33h		Path A 5G BW40-1S Power Index for Ch 44, 46, 48, Range 0~127.	50 h
34h		Path A 5G BW40-1S Power Index for Ch 52, 54, 56, Range 0~127.	50 h
35h		Path A 5G BW40-1S Power Index for Ch 60, 62, 64, Range 0~127.	50 h
36h		Path A 5G BW40-1S Power Index for Ch 100, 102, 104, Range 0~127.	50 h
37h		Path A 5G BW40-1S Power Index for Ch 108, 110, 112,Range 0~127.	50 h
38h		Path A 5G BW40-1S Power Index for Ch 116, 118, 120, Range 0~127.	50 h
39h		Path A 5G BW40-1S Power Index for Ch 124, 126, 128, Range 0~127.	50 h
3Ah		Path A 5G BW40-1S Power Index for Ch	50 h

		132, 134, 136,Range 0~127.	
3Bh		Path A 5G BW40-1S Power Index for Ch 140, 142, 144,Range 0~127.	50 h
3Ch		Path A 5G BW40-1S Power Index for Ch 149, 151, 153, Range 0~127.	50 h
3Dh		Path A 5G BW40-1S Power Index for Ch 157, 159, 161,Range 0~127.	50 h
3Eh		Path A 5G BW40-1S Power Index for Ch 165, 167, 169, Range 0~127.	50 h
3Fh		Path A 5G BW40-1S Power Index for Ch 173, 175, 177, Range 0~127.	50 h
40h	Path A 5G BW20-1S Tx Power Index Difference Path A 5G OFDM-1Tx Power Index Difference	Power Index Difference between BW20-1S and BW40-1S. Bit[7:4]: Path A 5G Offset, Range –8~7. (0.5dB/step) Power Index Difference between OFDM-1Tx and BW40-1S. Bit[3:0]: Path A 5G Offset, Range –8~7. (0.5dB/step)	02 h
41h~C7h	Reserved	Reserved for Realtek.	-
C8h	Channel Plan (reserved)	(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	40 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	66 h
CBh	IQ Calibration and LC Calibration	Reserved	00 h
CCh~CFh	Reserved	Reserved for Realtek.	-
11Ah~0x11Fh	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.	00E04C000001 h
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) 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	21 h

		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	FF 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	02h
135h	Customer ID	Customer ID (0x00 and 0xFF are reserved for Realtek)	FFh
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 PathA OFDM 0h: 0dB (default), 1h: -3dB, 2h: -6dB, 3h: -9dB	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 : HE40, MCS9 256QAM Bit[1]: 5G 40M Tx Power Calibrator Rate. 0h : HT40, MCS7 64QAM (default); 1h : HE40, MCS9 256QAM Bit[3:2]: reserved Power Tracking	00h

		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.	00h
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 worldwide ACSII chars table : https://www.asciicharstable.com/	30h
13Ch	Country code		30h
13Dh	Power Supply Mode	Bit[3:0] Power supply mode selection to determine which RF parameters should be loaded by driver 0 or Fh: Power Supply Mode 0, system DC power 3.3V, PA DC power 3.3V, normal mode 1h: Power Supply Mode 1, system DC power 1.8V, PA DC power 1.8V 2h: Power Supply Mode 2, system DC power 3.3V, PA DC power 1.25V 3h to Eh: Reserved Bit[7:4] Reserved	FFh
13Eh~14Dh	Reserved	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 = 0x00E04C800001:
		0x1B4=00
		0x1B5=E0
		0x1B6=4C
		0x1B7=80
		0x1B8=00
		0x1B9=01
			00E04C800001h

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)	10h

		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.	

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	3.3V: 3Fh , 1.8V:45h
1C5h	lbt_ant_gain	Bit[7:0]: lbt_ant_gain(default 0x7)	3.3V: 07h , 1.8V: 0h
1C6h	Reserved	Reserved for Realtek. Do not change this field without Realtek's approval.	-
1C7h	iqm_max_txgain_LE1M	1M Max Tx Gain	3.3V:3Bh , 1.8V:44h
1C8h	iqm_max_txgain_LE2M	2M Max Tx Gain	3.3V:3Bh , 1.8V:44h
1C9h-1CAh	Reserved	Reserved for Realtek. Do not change this field without Realtek's approval.	-
1CBh	iqm_max_txgain_LR	LR Max Tx Gain	3.3V:3Bh , 1.8V:44h
1CCh~1CFh	Reserved	Reserved for Realtek. Do not change this field without Realtek's approval.	-
1D0h	tmeterx4_txgaink_module	T meter value	66h
1D4h~1FFh	Reserved	Reserved for Realtek. Do not change this field without Realtek's approval.	-

Table 12-1: WLAN and BT Controller Calibration Data Contents

13 Appendix: WiFi 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.

14 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

asciicharstable.com