
Application Note : Telink BLE Mesh Development Kit And SDK Manual

AN-BLE-2015060300-E10

Ver 1.8.0

2016/5/30

Keywords:

Light Module; Switch; APP; SDK; Firmware; Compiling;
Burning; OTA; Master Dongle; Packet sniffer

Brief:

This document is the usage and development guide for
Telink BLE mesh development kit and SDK.



TELINK SEMICONDUCTOR

Published by
Telink Semiconductor

Bldg 3, 1500 Zuchongzhi Rd,
Zhangjiang Hi-Tech Park, Shanghai, China

© Telink Semiconductor
All Right Reserved

Legal Disclaimer

Telink Semiconductor reserves the right to make changes without further notice to any products herein to improve reliability, function or design. Telink Semiconductor disclaims any and all liability for any errors, inaccuracies or incompleteness contained herein or in any other disclosure relating to any product.

Telink Semiconductor does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights, nor the rights of others

The products shown herein are not designed for use in medical, life-saving, or life-sustaining applications. Customers using or selling Telink Semiconductor products not expressly indicated for use in such applications do so entirely at their own risk and agree to fully indemnify Telink Semiconductor for any damages arising or resulting from such use or sale.

Information:

For further information on the technology, product and business term, please contact Telink Semiconductor Company (www.telink-semi.com).

For sales or technical support, please send email to the address of:

telinknsales@telink-semi.com

telinknsupport@telink-semi.com

Revision History

Version	Major Changes	Date	Author
1.0	Initial release	2015/4	J.G.H., Cynthia
1.1.0	Updated Figure 2, 8~10, 28~29, and section 3, 4, 5.2~5.5, 5.7~5.11, 7; Added section 5.5, 5.12~5.13 and 8~9	2015/6	J.G.H., Cynthia
1.2.0	Replaced RC operation with Switch operation and illustrate FW compiling and burning operation further; Updated emulator tool operation on PC side; Updated explanations for source code, communication mechanism, advertising packet, service uuid, get light status, mesh and flash storage areas; Added "get online status"; Reserved APP operation and FW upgrade via Master USB dongle; Updated FW version management.	2015/8	J.G.H., Cynthia
1.3.0	Added "Schedule" function; Updated pre-defined command table for emulator tool; Added description of destination address allocation; Updated Mesh Opcodes and parameters table;	2015/9	S.Q.F., Cynthia

	Added "Factory Reset" section; Updated flash storage address table; Added a note for adv packet during APP development.		
1.3.1	Updated advertising packet figure; Added FW version information figure; Updated description of "OTA" character on APP side	2015/9	S.Q.F., Cynthia
1.4.0	Updated write MAC_ID and read MAC_ID commands; Updated pre-defined command table (Music_Start/Music_Stop, Set_R_0/255, Set_G_0/255, Set_B_0/255, Set_RGB, Set_CT, User_All), mesh opcodes and parameters table (0xD2, 0xEA~0xEB) for emulator tool operation; Added "light_onoff_step" function on light module side; Furtherly illustrate "Get online status" function; Updated figures for "Clock" and "Alarm Clock" function; Updated important flash storage address.	2015/10	S.Q.F., Cynthia
1.5.0	Updated clock function in schedule	2015/10	S.Q.F., Cynthia
1.6.0	Updated pre-defined command table for emulator tool operation: added Almon_Add/Almoff_Add/AlmSce_Add,	2015/12	Z.J.Y., J.G.H., Cynthia

	<p>Add_Scene/Del_Scene and Get_Scene commands; deleted Alarm_Add.</p> <p>Updated Mesh Opcodes and parameters table (0xC0~0xC1, 0xEC~0xEF) as well as notes for emulator tool operation;</p> <p>Added scene mode;</p> <p>Updated schedule function;</p> <p>Updated flash storage address table.</p>		
1.7.0	<p>Updated “advertising packet” figure;</p> <p>Added Light initiatively sending notify data;</p> <p>Updated description of OP “0xE3” in Mesh Opcodes and parameters table.</p>	2015/12	S.Q.F., Cynthia
1.8.0	<p>Updated Mesh Opcodes and parameters table (0xC0, 0xE6);</p> <p>Updated Burning Key related figures;</p> <p>Updated figure of “Configure MAC_ID”: write MAC_ID command.</p>	2016/5	S.Q.F., Cynthia

Table of contents

1	Introduction	9
1.1	Package Content	9
2	Firmware Compiling and Burning Operation	12
2.1	Compiling operation	12
2.2	Firmware burning.....	13
2.2.1	Firmware burning on BLE light module	13
2.2.2	Firmware burning on BLE switch	16
3	Emulator Tool Operation	17
3.1	Master Dongle.....	17
3.2	tl_ble_phone_mesh2.exe	17
3.3	Step-by-step user guide	20
3.4	command¶ms	21
4	Switch Operation	28
4.1	Configure switch	28
4.2	Button functions in standalone switch mode.....	28
5	APP operation.....	29
6	SDK Development.....	30
6.1	Overall architecture	30
6.2	Macro definition	31
6.3	Source code on light module side.....	32
6.4	Source code on switch side.....	35
6.5	Basic communication mechanism	36
6.5.1	Switch + Light Module	36
6.5.2	APP + Light Module.....	36
6.6	Advertising packet	37
6.7	Service uuid.....	38

6.8	Get online status	39
6.9	Get light status	41
6.10	Light initiatively sending notify data	42
6.11	Scene mode	42
6.11.1	Add/Modify scene mode	42
6.11.2	Load scene mode	45
6.11.3	Delete scene mode	46
6.12	Schedule	47
6.12.1	Clock	47
6.12.2	Alarm clock	48
6.13	Mesh	54
6.14	Flash storage areas	55
7	OTA	56
7.1	Use Master USB dongle as update tool	56
7.2	Use APP as update tool	56
7.3	OTA development description on APP side	59
8	Factory Reset	61
9	Firmware version management	62
10	Packet Sniffer	63
11	Reference Documents	64

Table of figures

Figure 1	BLE light module/Master USB dongle	10
Figure 2	Buttons of switch unit	10
Figure 3	Firmware burning EVK	11
Figure 4	Compiling operation	12
Figure 5	“Clean project” operation	12
Figure 6	BLE light module, EVK and PC connection	13
Figure 7	Light bin file burning example	14
Figure 8	Correct burning operation indication in log window	15
Figure 9	Configure MAC_ID	15
Figure 10	BLE Switch, EVK and PC connection	16
Figure 11	Tool GUI	17
Figure 12	Security information field	18
Figure 13	“All_on” command	25
Figure 14	General BLE Module/switch firmware flow	30
Figure 15	Light switch configuration example	35
Figure 16	Advertising packet	37
Figure 17	Attributes	38
Figure 18	Packet sniffer log 1: Light switch-on command	39
Figure 19	Packet sniffer log 2	39
Figure 20	Enable “get online status”	39
Figure 21	Online status data	40
Figure 22	BLE light module status	41
Figure 23	Packet sniffer result	41
Figure 24	User notify data	42
Figure 25	Add scene mode 01 for Light node 001 (device ID: 0013)	44
Figure 26	Add scene mode 01 and 02 to light node 001	44
Figure 27	Modify scene mode 02	45
Figure 28	Load scene mode 02	46
Figure 29	Delete scene mode 02	46
Figure 30	Clock function	47
Figure 31	Add alarm clock	50
Figure 32	Obtain light’s alarm clock information	51
Figure 33	Alarm clock status	51
Figure 34	Turn off alarm clock 2	52
Figure 35	Modify alarm clock 1	53
Figure 36	Delete alarm clock 3	54
Figure 37	Burn ble_master_dongle.bin and new firmware into update board	56
Figure 38	Burn old firmware and OTA boot loader into light board	57
Figure 39	OTA update flow	58
Figure 40	OTA update on APP side	59
Figure 41	OTA packet	60

Figure 42	FW version information	62
-----------	------------------------------	----

Table of tables

Table 1	Command description for tl_ble_phone_mesh2.exe.....	20
Table 2	Mesh Opcodes and parameters	22
Table 3	switch button functions.....	28
Table 4	Main macro definition	31
Table 5	Main functions of /vendor/light_8266/main_light.c	32
Table 6	Main functions of /proj_lib/light_ll/light_ll.h	34
Table 7	GPIO related main functions	34
Table 8	Main function of light_switch.c.....	35
Table 9	Attribute functions	38
Table 10	Important flash storage address.....	55
Table 11	u8 dat[23]	59

1 Introduction

This document provides guidance on using the Telink BLE mesh development kit, mainly describing firmware compiling and burning operation, switch usage guide, SDK development interface guide, OTA (Over The Air) upgrade guide, Master Dongle usage guide, and packet sniffer usage guide.

This document helps customers to get familiar with Telink BLE mesh development kit and SDK as soon as possible and quickly start value added development on this basis.

1.1 Package Content

The Telink BLE mesh development kit contains the following hardware and software tools:

- BLE light modules: As shown in **Figure 1**, this module emulates BLE light control board and it contains 512KB Flash, 2 buttons (SW1 and SW2), 4 LED lights, USB and Single Wire interface for programming and debugging. The lights' on/off, brightness, and RGB value can be controlled via Telink BLE switch.
- 1 switch unit: it has 14 buttons, 1 indicating LED, as shown in **Figure 2**.
- "Telink BLE Lighting" Android APP: TBD.
- 1 EVK board (Burning Key): As shown in left of **Figure 3**, it contains Flash, 8 LED lights, USB and Single Wire interface for programming and debugging. The EVK board is used as the adapter board for firmware burning.
- 1 OTA-Master dongle, as shown in **Figure 1**. The dongle can be used as update board for firmware update on the BLE light module through OTA.
- Telink IDE (Integrated Development Environment) and WtcdB: Development and programming tool on PC side integrated with Eclipse.
- Packet Sniffer: Packet capturing tool on PC side.

*Note: The BLE light module is the same as the master dongle on the hardware.

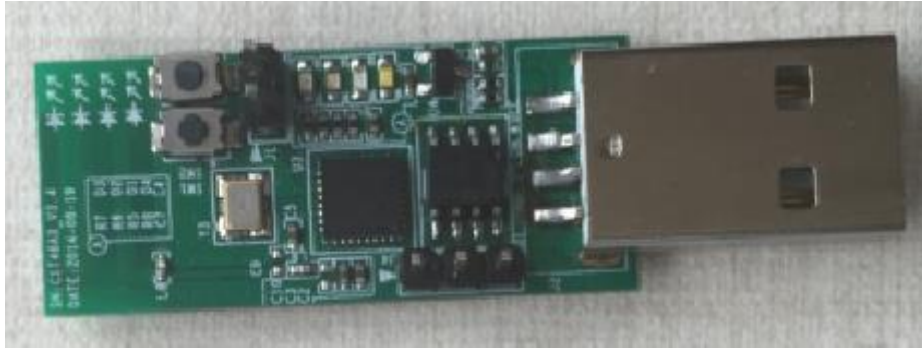


Figure 1 BLE light module/Master USB dongle




Figure 2 Buttons of switch unit



Figure 3 Burning Key

2 Firmware Compiling and Burning Operation

2.1 Compiling operation

After importing the project, click the “light_8266”/”light_8267” (depending on whether the BLE light board adopts TLSR8266 or TLSR8267)/”light_switch_8266” option under the  icon to carry out automatic compiling operation for corresponding BLE light and switch module source code branch.

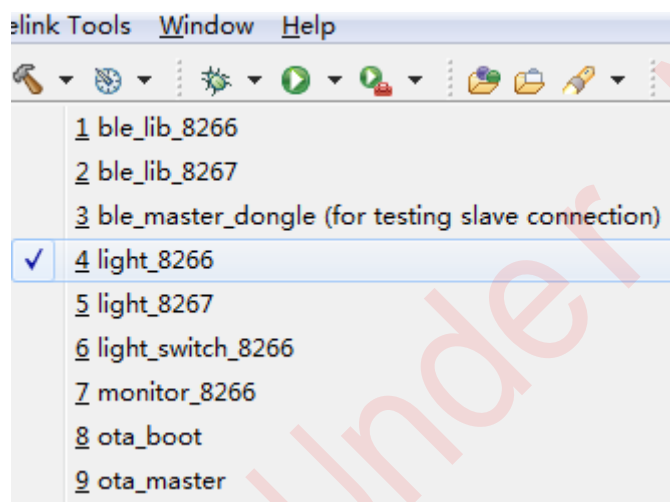


Figure 4 Compiling operation

During development, if only header file is modified, a “Clean Project” operation is needed before re-compiling.

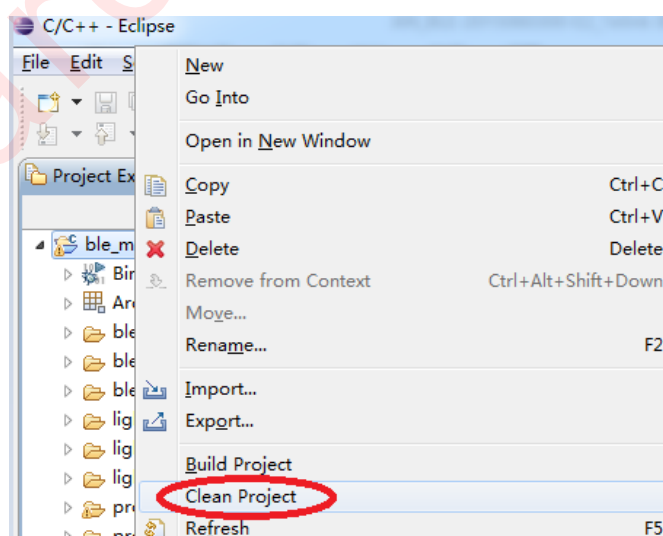


Figure 5 “Clean project” operation

2.2 Firmware burning

2.2.1 Firmware burning on BLE light module

The EVK and WtcdB tool are used to burn the light bin file (e.g. light_8266.bin) into the BLE light module (e.g. light module based on TLSR8266) according to steps below:

- 1) Connect hardware: Connect the EVK with PC via an USB cable. The indicating light on the EVK will twinkle once to indicate that the EVK board and its connection with PC is OK. Then connect BLE light module with the EVK via USB interface.



Figure 6 BLE light module, EVK and PC connection

- 2) Open the WtcdB tool on PC side, and select "**wt_light.ini**" in the drop-down box as shown in mark 1 of **Figure 7**.
- 3) Click the "**BIN**" button (as shown in mark 2 of **Figure 7**) and select the directory containing the light_8266.bin (as shown in mark 3 of **Figure 7**). All BIN files (including the light_8266.bin) within the directory will be available in the left window.
- 4) Select the "light_8266.bin" in the left window, as shown in mark 4 of **Figure 7**.
- 5) Click the "RstMCU" button (as shown in mark 5 of **Figure 7**) to reset MCU of the BLE light module.

- 6) If this is the first time the module is being programmed, click the “EraseF” button (as shown in mark 6 of **Figure 7**) to erase the Flash of the BLE light module. If this is not the initial burning and various ID information of the light module need to be retained, this step should be skipped.
- 7) Click the “SWB” button (as shown in mark 7 of **Figure 7**) to download the light_8266.bin into **Flash** of the BLE light module.

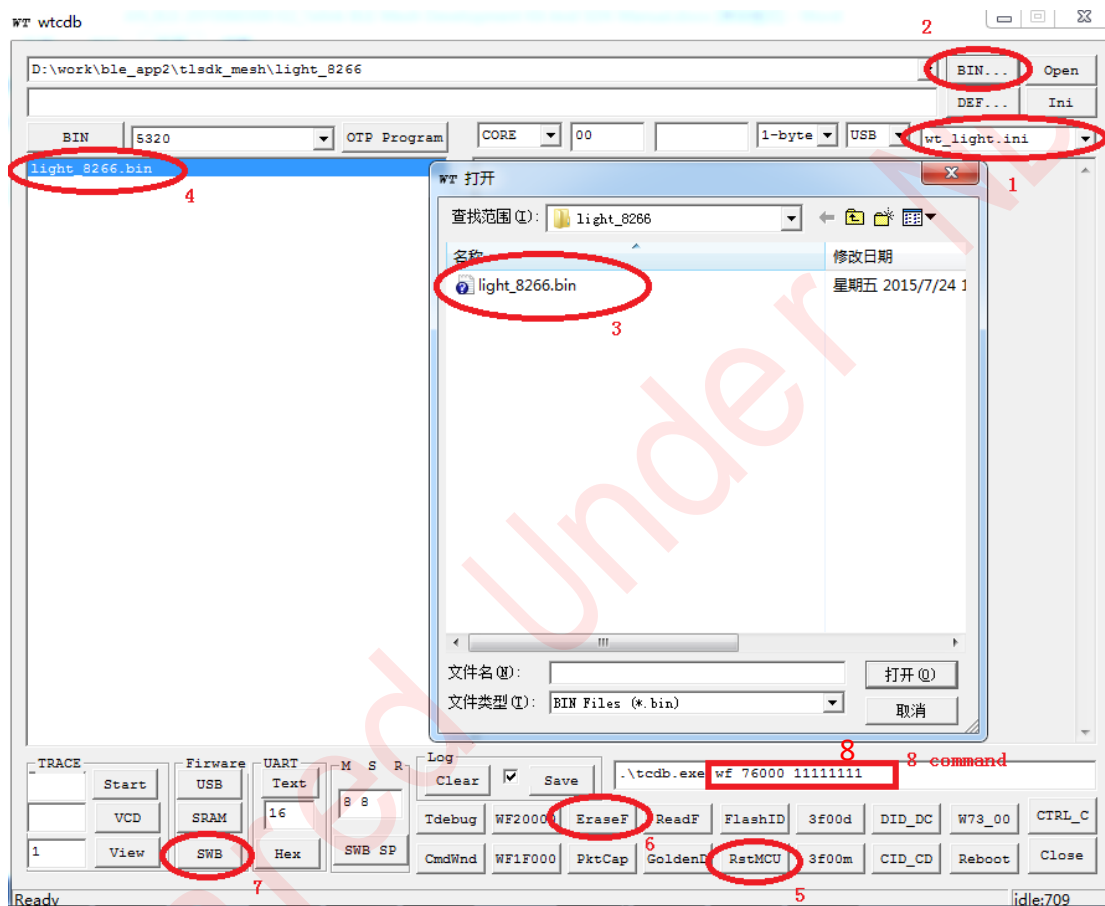


Figure 7 Light bin file burning example


```

.\tcd.db.exe rst
Slave MCU Reset 5

Flash Sector (4K) Erase at address 3d000
Flash Sector (4K) Erase at address 3e000
Flash Sector (4K) Erase at address 3f000
Total Time: 1865 ms 6

tcd.db -i light_8266.bin -b
TC32 EVK: Swire OK
Flash Sector (4K) Erase & Program at address 0
Flash Sector (4K) Erase & Program at address 1000
Flash Sector (4K) Erase & Program at address 2000
Flash Sector (4K) Erase & Program at address 3000
Flash Sector (4K) Erase & Program at address 4000
Flash Sector (4K) Erase & Program at address 5000
Flash Sector (4K) Erase & Program at address 6000
file download to 00000000: 26904 bytes 7
Total Time: 1751 ms

```

Figure 8 Correct burning operation indication in log window

8) Configure MAC_ID manually:

Successively enter corresponding write MAC_ID and read MAC_ID commands in the command window (as shown in mark 8 of **Figure 7**), and click the “Enter” key to execute the command.

```

.\tcd.db.exe wf 76000 11111111 write MAC_ID
TC32 EVK: Swire OK
Flash Sector (4K) Program at address 76000
Total Time: 5 ms
.\tcd.db.exe rf 76000 -s 16 read MAC_ID
TC32 EVK: Swire OK
0000 11 11 11 11 ff ff ff ff ff ff ff ff ff ff ff ff
Total Time: 2 ms

```

Figure 9 Configure MAC_ID

Repeat the above steps to burn multiple BLE light modules. The MAC ID of different light modules needs to be unique.

2.2.2 Firmware burning on BLE switch

Firmware burning method of BLE switch is similar to the method of BLE light module, except for the following:

- 1) The hardware connection: The BLE switch is connected with EVK via Swire interface.

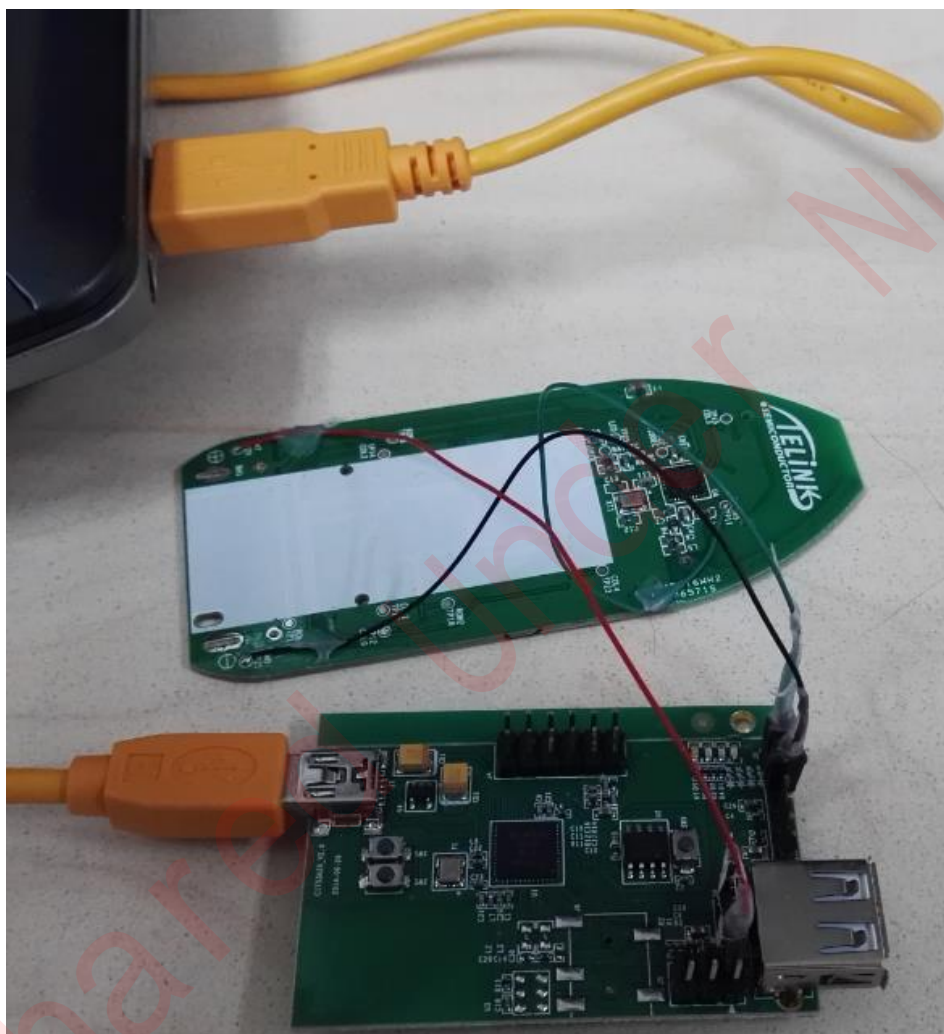


Figure 10 BLE Switch, EVK and PC connection

- 2) The firmware to be burned is named “light_sw_8266.bin” by the auto build tool.

*Note: When the BLE switch enters sleep state, the burning operation may be unsuccessful as the EVK fails to connect with the switch. Under this situation, simply press any button to wake up the switch to work state and use “RstMCU” to reset the switch so that it will not enter sleep state before burning starts.

3 Emulator Tool Operation

One Master Dongle and BLE Mesh Emulator tool (tl_ble_phone_mesh2.exe) on PC side are needed to emulate some of the control that can be performed on a mobile phone.

Master Dongle connected with PC USB interface can establish connection and communicate with the light module. It can be used for early debugging when developing APP on the phone.

The BLE Mesh Emulator tool allows the developer to connect to BLE mesh nodes (light modules) using the Master Dongle and emulates security information setting, group assigning, node status controlling and polling operations.

3.1 Master Dongle

- 1) Download the ble_master_dongle.bin into the dongle via the EVK board and wtcdm tool.
- 2) Connect the master dongle with PC via USB interface.
- 3) Power on the light module.

3.2 tl_ble_phone_mesh2.exe

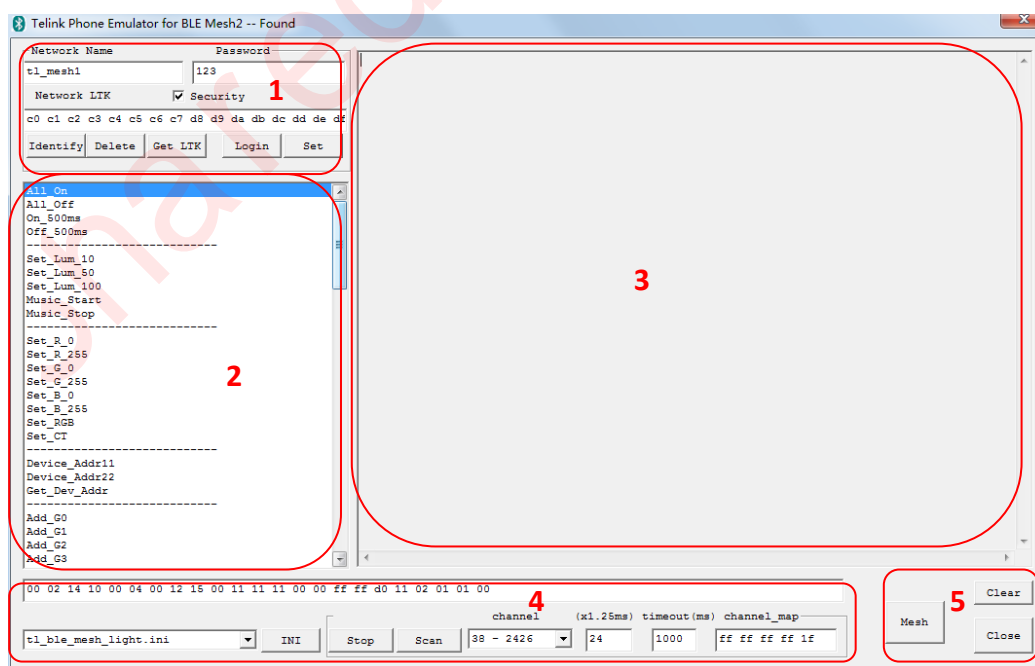


Figure 11 Tool GUI

The Emulator tool GUI is shown in Figure 11, with several main components included. The first section “1” show the security information related to the mesh network, including information such as network name, password for accessing the network, and the network long term key. A number of control buttons are also included to allow operations related to these security fields.

The second section “2” includes a series of pre-defined control commands that can be used to control all mesh nodes. For example, commands for group assignment, commands for adjusting RGB, commands for turning nodes on/off, and commands for querying current node status.

The third section “3” is a log window, it shows in real time the related log information for the mesh network or the connected mesh nodes, for example, details in commands which before and after encryption and decryption, failure/successful status of commands.

The fourth section “4” also contains a read-only text box for showing the details of the pre-defined commands, text box showing general physical layer information such as the channel used and the connection interval settings. There are also a number of buttons to allow scan, connect and disconnect to a mesh node. All the pre-defined commands are configurable in a text configuration file “tlk_ble_mesh_light.ini” and can be opened and edited by clicking “INI” button.

The fifth section “5” contains a “Mesh” button to open user control interface, a “Clear” button to clear the log window, and a “Close” button to close the tool interface.

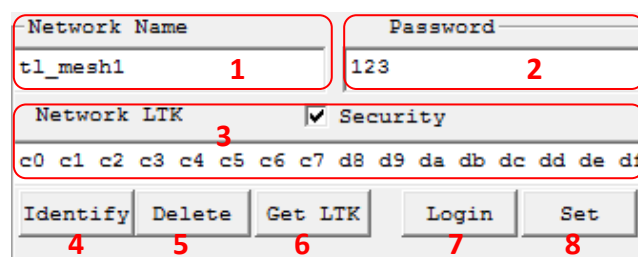


Figure 12 Security information field

Within the first section, button “Identify” is used to find a mesh node that is in

factory default state and has not been added to any mesh network;

Button “Delete” is used to delete a mesh node from the current mesh network;

Button “Get LTK” reads the LTK information of the mesh.

Button “Login” is used to confirm the network name and password information with a mesh node and establishes a security association before the Emulator tool can change any of the these values on the mesh node;

Button “Set” is used to set new values for network name, password, or network LTK on a logged mesh node.

3.3 Step-by-step user guide

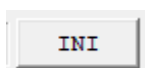
Please refer to the document “**AN_BLE-15050400_User Guide For Telink BLE Mesh Emulator tool**” for details about step-by-step user guide.

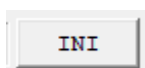
- 1) Pre-defined Command description is shown as in the Table below. (Double clicking pre-defined command icon is to execute corresponding command.)

Table 1 Command description for tl_ble_phone_mesh2.exe

Command	Function
All_on/All_Off	Switch on/off all lights
On_500ms/Off_500ms	Delay 500ms, on/off
Set_Lum_10/50/100	RGB: 10%,50%,100% luminance
Music_Start/Music_Stop	Start/Exit Music mode
Set_R_0/255	R: color index
Set_G_0/255	G: color index
Set_B_0/255	B: color index
Set_RGB	RGB: color index
Set_CT	CT: Color Temperature
Device_Addr11/22	Set device address as 0x1111/0x2222
Get_Dev_Addr	Obtain device address
Add_G0/Add_G1/...	Assign light to Group1/2/...
Del_G0/Del_G1/...	Remove Group1/2/...
Kick_out	Kick the node out of mesh
Get_G_8	Query for short group ID
Get_G_F4	Query for long group ID of the first four groups
Get_G_B4	Query for long group ID of the last four groups
G1_On/G1_Off	Switch on/off lights of Group1
User_All	Customize return value for notification from light node
Status_All	Obtain all light's status
Status_G1/2/3/4	Obtain lights status of Group 1/2/3/4
SW_config	Configure switch
Set button	Set name, pwd and ltk for currently connected slave
Time_Set	Set time information
Time_Get	Read time information
Get_Alarm	Read alarm clock information
Almon_Add/Almoff_Add/AlmSce_Add	Add alarm clock with light on/light off/scene mode event
Alarm_Del/Alarm_Chg Alarm_En/Alarm_Dis	Delete/Modify/Open/Close alarm clock
Add_Scene/Del_Scene	Add/Delete scene mode
Get_Scene	Read information of scene mode

***Note:**



User can click the  icon to open the tl_ble_mesh_light.ini file and modify pre-defined command as required. Please refer to **Section 3.4** for command format.

3.4 command¶ms

Main data structure is shown as below:

```
typedef struct{
    u32 dma_len;
    u8  type;
    u8  rf_len;
    u16 l2capLen;
    u16 chanId;
    u8  opcode;
    u8  handle;
    u8  handle1;
    u8  value[30]; //sno[3],src[2],dst[2],op[1~3],params[0~10],rsv[10]
} rf_packet_att_cmd_t;

    sno[3]:need to be unique

    src[2]:only be used in mesh packet

    dst[2]: destination address
```

*Note: The destination address allocation is illustrated as below.

- 1) 0000000000000000 - LocalAddress: Indicate the light(s) directly connected with APP.
- 2) 1111111111111111 - BroadcastAddress: All lights will respond if the destination is broadcast address 0xffff.
- 3) Device / Group addresses differ by bit[15].
 - ✧ 0xxxxxxxxxxxxxx - DeviceAddress: Device address, ranging from 0x0001 to 0x7fff.
 - ✧ 1xxxxxxxxxxxxxx - GroupAddress: Group number address, ranging from 0x8000 to 0xfffe.

All the commands to and from the mesh network contain 1~3 bytes of Opcode and 0~10 bytes of parameters. The currently defined opcode and parameters are specified in Table 2, only 3-byte opcode is used for now and the opcode takes the format of 11xxxxxxzzzzzzzzzzzzzzzzzzzz, where “zzzzzzzz” is the two-byte company id

(0x0211 for Telink). All opcodes and parameters in Table 2 use handle of the Mesh light command except for 0xD4/D5/D6/DB, which use the handle corresponding to the Mesh light status attribute.

Table 2 Mesh Opcodes and parameters

OP	xxxxxx(bit)	Description (length and meaning of bytes in params field)	
0xC0	000000(0x00)	Obtain scene parameters. Params[0]: relay count. Params[1]: scene ID. 00 means reading all id.	
0xC1	000001(0x01)	Scene information response	Refer to Note 4 for parameter format
0xD0	010000(0x10)	Params[0]=1, light on Params[0]=0, light off	
0xD1	010001(0x11)	Reserved	
0xD2	010010(0x12)	1) If Params[0] is among the range of 5~100: Set brightness, Params[0] identifies the percentage of lumen. 2) If Params[0] equals 0xFE, it's used as "MUSIC_START" command, and the indicating lights (all nodes within the network) store current luminance. Volume value for the music to be played can be sent via normal luminance command. 3) If Params[0] equals 0xFF, it's used as "MUSIC_STOP" command, and the indicating lights restore the luminance stored during "MUSIC_START".	If no volume data has been received when the pre-configured timeout duration (e.g. 3s) expires, the nodes exit music mode and restore to normal mode.
0xD3	010011(0x13)	Demo code for config switch.	
0xD4	010100	Response to short group ID query, Params[0-7] is the lower bytes of the 8 groups the light node belongs to, 0xFF indicates a null group	Each light node can be assigned to a maximum of 8 groups.
0xD5	010101	Response to long group ID query, Params[0-7] is the group ID for the first four groups the light node belongs to, 0xFFFF indicates a null group	
0xD6	010110	Response to long group ID query, Params[0-7] is the group ID for the last four groups the light node belongs to, 0xFFFF indicates a null group	
0xD7	010111	Params[0]=1, add group Params[0]=0, remove group. If params[1-2]=0xFFFF, then	Params[1~2], group id.

OP	xxxxxx(bit)	Description (length and meaning of bytes in params field)			
		kick the node out of mesh.			
0xD8	011000	Reserved			
0xD9	011001	Reserved			
0xDA	011010	Status query, Params[0]: relay count			
0xDB	011011	Status response, Params[0~5] the PWM setting of the light node, Params[6-7] Reserved, Params[8]=TTC, Params[9]=hops			
0xDC	011100	Enable the function of online status report.			
0xDD	011101	Group ID query, Params[0] is the relay count	Params[1]=1, query for short group ID (the lower bytes of the group IDs for up to 8 groups)		
			Params[1]=2, query for long group ID of the first four groups		
			Params[1]=3, query for long group ID of the last four groups		
0xDE	011110	Be used for the left/right key function of switch.			
0xDF	011111				
0xE0	100000	Set device address, Params[0-1] is the new device address. If never configured, it will use the low 2bytes of mac address. When one light receives this command and responds to it, it will actively notify the master with its own address. *Note: If Params[0-1] is 0xffff, this command won't set address for it's invalid always, but can obtain device addr of all lights indicated by the destination address (generally it's group number). That is, this command can be used to obtain all lights' device addr within a group.			
0xE1	100001	Notify the device address information, Params[0~1] is the device address.			
0xE2	100010	Params[0]=1	Set R's lumen.	Params[1]: the percentage of	

OP	xxxxxx(bit)	Description (length and meaning of bytes in params field)			
		Params[0]=2	Set G's lumen.	lumen	
		Params[0]=3	Set B's lumen.		
0xE3	100011	Kick a node out of mesh. Function: 1. Restore to factory default setting. 2. Configure mesh name according to Params[0]. Params[0]: Without this parameter, or if it's set as 0, the mesh name of light will be set as "out_of_mesh"; if the parameter is set as "1", the mesh name of light will be set as the initial value "telink_mesh1".			
0xE4	100100	Set time information: Params[0~1]: Year; Params[2]: Month; Params[3]: Day; Params[4]: Hour; Params[5]: Minute; Params[6]: Second.			
0xE5	100101	Params[0]=0	Add alarm clock	Refer to Note 1 for parameter format	
		Params[0]=1	Delete alarm clock		
		Params[0]=2	Modify alarm clock		
		Params[0]=3	Open alarm clock		
		Params[0]=4	Close alarm clock		
0xE6	100110	Obtain alarm clock parameters. Params[0]: relay count. Params[1]: alarm ID. 00 means reading all id.			
0xE7	100111	Alarm information response			Refer to Note 2 for parameter format
0xE8	101000	Read time information. Params[0]: relay count.			
0xE9	101001	Time response. Params[0~1]: Year; Params[2]: Month; Params[3]: Day; Params[4]: Hour; Params[5]: Minute; Params[6]: Second;			

OP	xxxxxx(bit)	Description (length and meaning of bytes in params field)	
		Params[7:9]: 0xfffff (reserved)	
0xEA	101010	Customize return value for the notification from light node	
0xEB	101011	Response for the "Customize notification return value" command. Up to 10-byte customized data can be returned: Params[0~9].	
0xEC	101100	reserved	
0xED	101101	Set RGB values of lights (for remote use only) Params[0]: index of table "rgb_map". Please refer to main_light.c for "rgb_map".	
0xEE	101110	Params[0]=0: Delete scene mode. params[1]: ID of scene mode. If ID is "0xff", it indicates all scene modes are to be deleted.	Please refer to Note 3 for scene mode data.
		Params[0]=1: Add scene mode. params[1~5]: scene mode data.	
0xEF	101111	Load scene params[0]: scene ID	

The command of "All_on" is shown as an example:

L2CAP Header		ATT_Write_Req										CRC						
L2CAP-Length	ChanId	Opcode	AttHandle	AttValue														
0x0010	0x0004	0x12	0x0015	11	11	11	00	00	FF	FF	D0	11	02	01	01	00	0x961183	
				sno	src	dst	op	params										
				value[]														

Figure 13 "All_on" command

***Note1:**

- Params[1]: Alarm clock index (idx) number.
- Params[2]: bit combination parameter, as defined in alarm_ev_t.par1 of the code.
 - bit0~bit3: The four bits specify light response event when the alarm clock expires. Two types of event are defined: 0 indicates "light off", 1 indicates "light on", while 2 indicates "scene".

- ✧ bit4~bit6: The three bits specify alarm clock type. Two types of alarm clock are defined: 0 indicates “DAY” type, while 1 indicates “WEEK” type.
- ✧ Bit7: Alarm clock enabling flag. 1 indicates “enable”. For the “Alarm_Add” command, this bit should be set to “1” by APP to open the added alarm clock by default.

3) Params[3]:

- a) For “DAY” type alarm clock, it indicates “month” information ranging from 1 to 12 (corresponding to January~ December).
- b) For “WEEK” type alarm clock, it’s a reserved byte.

4) Params[4]:

- a) For “DAY” type alarm clock, it indicates “day” information.
- b) For “WEEK” type alarm clock, it indicates “week” information: bit0~bit6 indicates Sunday, Monday,, Saturday, respectively; bit7 is a reserved bit and should be set to “0” always to be legal.

5) Params[5]: Hour;

6) Params[6]: Minute;

7) Params[7]: Second;

8) Params[8]: Scene ID.

***Note2:**

- 1) Params[0]: Valid alarm clock flag. “0xa5” indicates “valid”, while “0” indicates “invalid”.
- 2) Params[1:7]: refer to Params[1:7] in **Note1**.
- 3) Params[8]: If the event triggered by the alarm clock is scene mode, params[8] indicates scene ID; otherwise params[8] should be 0.
- 4) Params[9]: This parameter indicates the alarm clock number for the light module. If the notification number received from alarm clocks does not equal this parameter value, the APP should resend a “0xe6” command.

When Params[0:8] are all "0x00", it indicates there's no alarm clock data for the light module at present.

***Note3:**

Scene mode data structure is as shown below:

```
typedef struct{
    u8 id;      //scene data index number
    u8 lum;     //luminance value
    u8 rgb[3];  // RGB values of lights
    u8 rsv[3];  //reserved data ("Set scene" does not transmit this parameter)
}scene_t;
```

***Note4:**

- 1) Params[0~7]: The data structure is same as scene mode data. Please refer to Note3.
- 2) Params[8]: Scene amount of the light.
- 3) Params[9]: reserved byte, should be 0xff.

4 Switch Operation

4.1 Configure switch

Any time after the switch unit is powered on, keep the “1_off” and “all_on” buttons pressed for one second, then release them. The indicating LED on the switch starts twinkling to indicate that the switch has entered the adv-mode. We can scan, connect, and configure its MESH_INFO via emulator tool then.

***Note:** MESH_INFO consists of MESH_NAME, MESH_PWD and LTK. The switch’s MESH_INFO must be the same as light’s MESH_INFO to control the lights.

4.2 Button functions in standalone switch mode

The switch can control all light modules with the same MESH_INFO.

The button functions on the switch unit are described in the table below:

Table 3 switch button functions

Button	Function	Note
all_on	Switch on all lights within the mesh	
all_off	Switch off all lights within the mesh	
1_on	Switch on lights of Group 1	
1_off	Switch off lights of Group 1	
2_on	Switch on lights of Group 2	
2_off	Switch off lights of Group 2	
3_on	Switch on lights of Group 3	
3_off	Switch off lights of Group 3	
4_on	Switch on lights of Group 4	
4_off	Switch off lights of Group 4	
Up	Increase luminance for lights which are recently switched on	After luminance is regulated for five seconds, current value will be stored automatically. e.g. If 1-on and 2_on are pressed, the “Up”/”Down” button is used to regulate luminance for Group 1 and Group 2. If all_on is pressed later, the “Up”/”Down” button is used to regulate all lights.
Down	Decrease luminance for lights which are recently switched on	
Left	Regulate attribute value for RGB or CT light	
Right		

5 APP operation

Please refer to Telink documents “***AN_BLE-16042500_Development Guide for BLE Mesh Lighting APP with Security***” and “***AN_BLE-16021900_User Guide For Telink tLight APP***” for BLE Mesh Lighting APP development and using guide, respectively.

6 SDK Development

6.1 Overall architecture

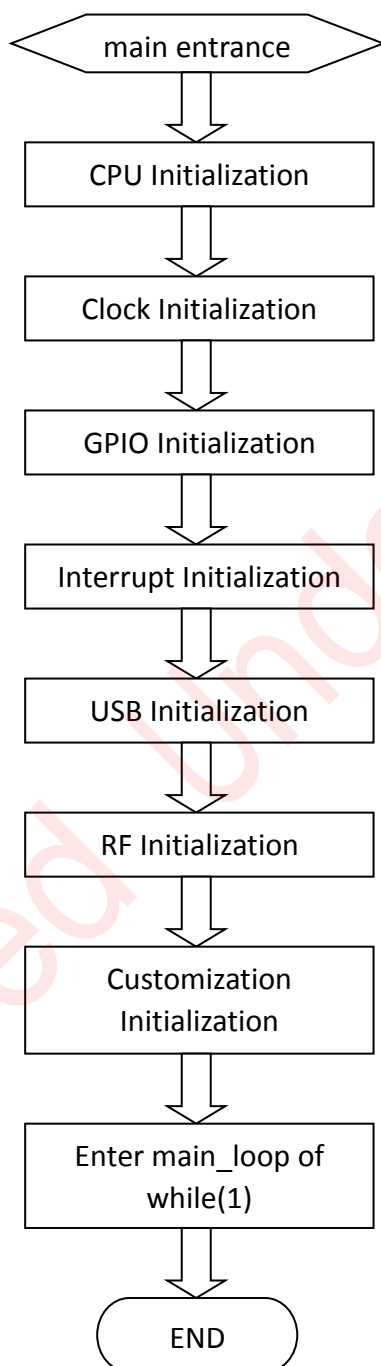


Figure 14 General BLE Module/switch firmware flow

6.2 Macro definition

Table 4 Main macro definition

Macro Name	Description	Note
PANEL_ENABLE	Enable/disable control panel function.	Disabled by default
LIGHT_MODE	RGB or CT mode	RGB mode by default
DEVICE_NAME	Device default name	
MESH_NAME	Mesh default name	The name will be returned during Master scan.
MESH_PWD	Mesh default password	
CLOCK_SYS_CLOCK_HZ	The frequency of clock.	CLOCK_SYS_CLOCK_HZ divided by
PMW_MAX_TICK	The ticks of one PWM cycle.	PMW_MAX_TICK is the frequency of PWM.
PMW_MAX_TICK_BASE	The base ticks of one PWM cycle.	
PMW_MAX_TICK_MULTI	The multiple of base ticks.	
BRIDGE_MAX_CNT	Maximum bridge times	
TELINK_SPP_UUID_SERVICE	Service UUID default value	
PWM_R/PWMID_R	PWM used for R light	
PWM_G/PWMID_G	PWM used for G light	
PWM_B/PWMID_B	PWM used for B light	
IRQ_TIMER1_ENABLE	Hardware timer enable flag	
IRQ_TIME1_INTERVAL	Hardware timer interval	
IC_8266F512	Whether TLSR8266F512 (with internal flash) is used for the switch	
rand_fix_flag	If the flag is 0, pair_rands is random number; If the flag is 1, pair_rands[8] = {0xb0, 0xb1, 0xb2, 0xb3, 0xb4, 0xb5, 0xb6, 0xb7}.	
not_need_login	1: Master does not need to login and can directly send commands to control lights.	

6.3 Source code on light module side

Table 5 Main functions of /vendor/light_8266/main_light.c

Function Name	Function Description	
user_init	User initialization, carry out some user-specific configuration.	
	Calling function	Description
	set_vendor_function	Some callback function can be defined in it
	light_init_default	Default value configuration
	gpio_set_func	Set pin as GPIO or PWM mode
	pwm_set	Set frequency and duty cycle for PWM pin
	light_lum_retrieve	Get stored lumen information and bring into effect
	pwm_start	PWM enable
	rf_link_slave_init	BLE protocol stack initialization, obtain MAC_ID/MESH_NAME/GRP_ID, set advertising packet content, etc.
	mesh_security_enable	Enable encryption function
	vendor_set_adv_data	Set adv-pkt's manufacturer data and the scan-response data.
	Other calling function	
	rf_link_set_debug_adv_channel	Set advertising channel, just used for debugging, should be commented for release version.
	rf_set_power_level_index	Set RF Tx power
	rf_link_set_max_bridge	Set maximum bridge times. Default: 8
	rf_link_set_max_relay	Set maximum relaying hops of light module after receiving non-APP command. Not yet called. Default: 3
	light_hw_timer1_config	Configure hardware timer
rf_link_data_callback	<p>After light module receives valid data, the lower layer will invoke this function, and respond to the received command (e.g. light switch on/off command)</p> <p>Command is defined in light_ll.h (e.g. LGT_CMD_LIGHT_ONOFF). User can extend customized command as required and respond to corresponding command in this callback function.</p> <p>Refer to Section 3.4 for data structure rf_packet_att_cmd_t.</p>	

Function Name	Function Description
rf_link_response_callback	The callback function is to respond to commands in the form of notify.
light_slave_tx_command	To realize communication between slaves.
light_lum_retrieve	Obtain luminance information
light_lum_store	Store luminance information
light_lum_erase	Erase luminance information
light_set_master_data	If any data needs to be sent to master, fill data in send_to_master[]; after master sends "get request", light module will respond to master with the data.
light_auth_check	In encryption mode, if user fails to authenticate password after schedule time expires, it will exit the mode automatically (not take effect yet).
light_user_func	User-defined functions.
light_sim_light2light	This function will send on/off command with interval of 5 seconds.
light_sim_notify	This function will send one notify data with interval of 5 seconds. Disabled by default.
proc_suspend	Suspend function. Disabled by default.
irq_handler	Interrupt handler function, including Rx/Tx interrupt, timer interrupt, GPIO interrupt, etc.
factory_reset_handle	Restore factory settings.
light_onoff_step	<p>Light gradual on/off function based on current RGB values: Decrease light luminance gradually until it's turned off, or turn on light node and gradually increase its luminance to specified value. Light adjustment period is 20ms.</p> <p>Two modes are supported:</p> <ol style="list-style-type: none"> 1) TYPE_FIX_STEP: Light luminance is adjusted every 20ms with fixed step (2). For example, to turn on light and gradually increase luminance to final value 100 (0→100), 50 (100/2) periods (i.e. 50*20ms=1s) are needed. To turn on light gradually from 0 to 50, 25 (50/2) periods (i.e. 25*20ms=0.5s) are needed. 2) TYPE_FIX_TIME: Default mode. Light luminance is adjusted to final value in fixed time (1s, i.e. 1s/20ms=50 periods). For example, to turn on light gradually from 0 to 100, light luminance is adjusted every 20ms with step of 2 (100/50). To turn on light gradually from 0 to 50, light luminance is adjusted with step of 1 (50/50).

*Note: RD staff should pay more attention to #if 1...#endif code.

Table 6 Main functions of /proj_lib/light_ll/light_ll.h

Function Name	Function Description	
light_set_tick_per_us	Set clock tick number corresponding to every microsecond	
rf_link_slave_init	Slave protocol stack initialization; parameter is interrupt interval responding to listening with unit of us. Enter adv_pkt sending state after responding for eight times.	
rf_link_slave_proc	Used by light module. It's used to judge timeout time during pairing process.	
rf_link_slave_set_adv	Set advertising packet (adv_pkt) content.	Set Flags-connectable, adv_data_type=0x01
rf_link_slave_set_adv_mesh_name		Set MESH name. Defined as MESH_NAME by default, adv_data_type=0x09.
rf_link_slave_set_adv_private_data		Set private information adv_data_type=0xFF. The content is data structure ll_adv_private_t; The content is data structure adv_rsp_pri_data for adv-rsp-pkt.
rf_link_light_event_callback	For some event's callback.	

Table 7 GPIO related main functions

Function Name	Function Description
gpio_set_output_en	Set pin output attribute. E.g. gpio_set_output_en (GPIO_PA4,1); indicates enabling GPIO_PA4 output.
gpio_set_input_en	Set pin input attribute.
gpio_write	Set pin as high/low level. E.g. gpio_write (GPIO_PA4,1); indicates setting GPIO_PA4 as high level.
gpio_set_func	Set pin as GPIO or PWM mode 0: GPIO Others: PWM
pwm_set	Set frequency and duty cycle for PWM pin.
pwm_start	PWM enable.

6.4 Source code on switch side

Under the “vendor/light_switch/” directory, Table 8 lists main functions of light_switch.c.

Table 8 Main function of light_switch.c

Function name	Description	
user_init	User initialization, carry out some user-specific configuration. 【Important function】	
	Calling function	Description
	cpu_set_gpio_wakeup	Configure GPIO pins for button wakeup
	rf_set_power_level_index	Set RF Tx power
	rf_link_set_debug_adv_channel	Set advertising channel
	mesh_security_enable	Enable encryption function
	adc_user_init	Adc initialization, used for low-voltage detect
proc_keyboard	Key scan, and send corresponding control command depending on scanning result. 【Important function】	
proc_suspend	Suspend function	
light_slave_tx_command	Send a command¶ms packet to light.	

Similar to light module, switch acts in normal mode by default. After configuring MESH_INFO for the switch via master dongle, all lights within the same mesh can be controlled (on/off) by the switch. The switch will enter sleep state after one second of no action, and can be wakened up by any button press.

The switch will enter configuration mode via pressing **all_on** and **1_off** buttons, and exit the mode via pressing any button. The configuration mode timeout duration is set as one minute. In configuration mode, the LED keeps blinking.

Master can configure switch as required. For example, the LED will blink for four times after the switch receives the command as below: ‘d3 11 02’ is cmd code, and ‘04’ is blink times:

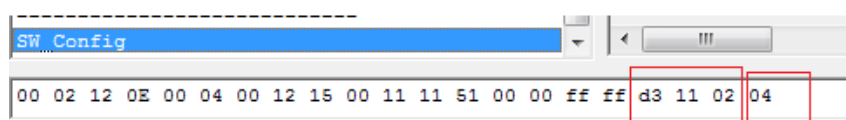


Figure 15 Light switch configuration example

6.5 Basic communication mechanism

6.5.1 Switch + Light Module

Connection is not established between BLE switch and light module. Any light module within receiving range can directly receive control command from the switch and relay it; the light module will only react on commands addressed to itself or to its group or to broadcast.

A light can be assigned to a maximum of 8 groups.

6.5.2 APP + Light Module

APP (Phone/Pad) will establish connection with light module selected by the user. All control commands will only be sent to this connected light module, and control function to other light modules can be realized by relaying the command through this connected light module.

Bridge connection number after the light module received APP command is defined as 8 by default, meaning the connected light module will repeat the command 8 times to ensure other light modules can receive it.

Relaying number after the light module received non-APP direct command is defined as 3 by default, meaning the light control command or status request command from non-App originator (i.e. not directly from a mobile phone) will only be relayed 3 hops at maximum.

So when node is far away from each other, control command is relayed according to below:

- 1) APP->light1->light2->light3->light4->light5;
- 2) switch->light1->light2->light3->light4.

Typical delay for each hop should be within 10ms.

6.6 Advertising packet

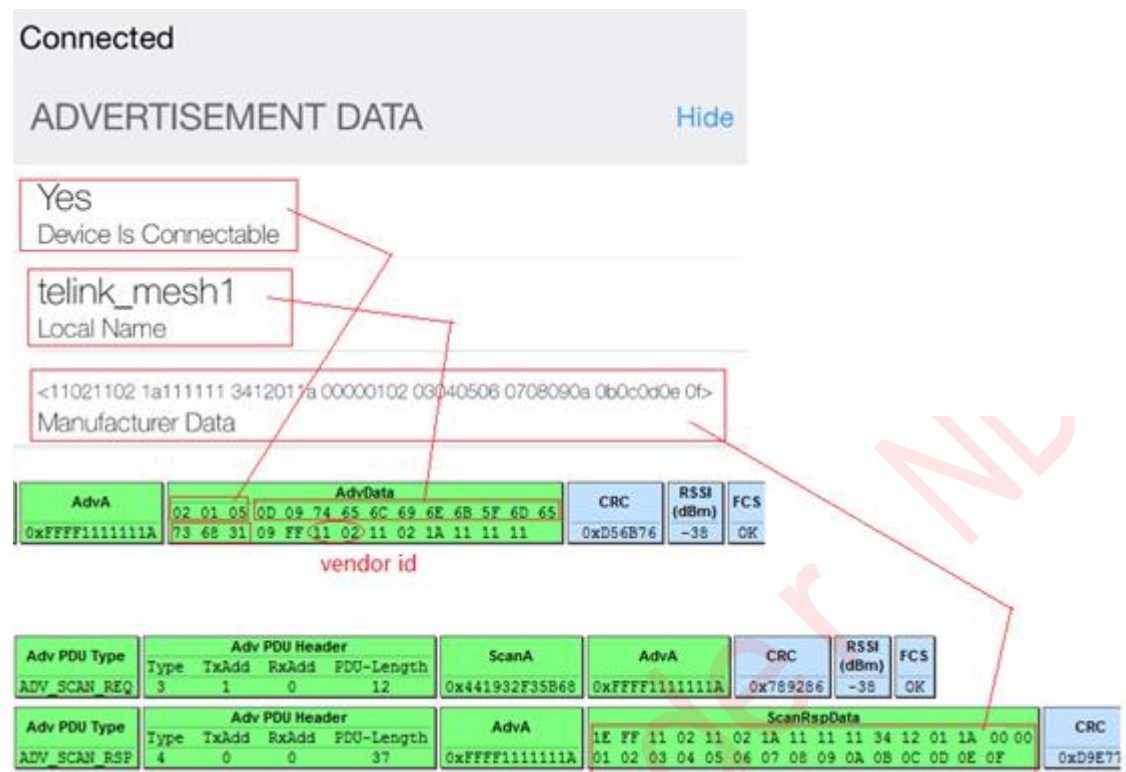


Figure 16 Advertising packet

Use the function of `vendor_set_adv_data` to assemble the advertising packet. Refer to Table 5 in **Section 6.3**.

*Note: During APP development, it's needed to filter out the advertising packets containing non-predefined vendor ID in the BLE peripheral scan list. The vendor ID value can be obtained from the "adv_pri_data" field of the advertising packet. As shown in the Figure 16, the vendor ID is 0x0211.

6.7 Service uuid

1) The light module currently defines four characters under service 0x000102030405060708090A0B0C0D1910, as shown below:



Figure 17 Attributes

2) Main functions for the attributes are shown as below:

Table 9 Attribute functions

character	Function	Sniffer data
Status	Slave (light) sends notify data to master (phone).	Refer to “get light status” in Section 6.9
Command	Master sends control command to light module, e.g. switch on/off command	See Packet sniffer log 1 below, params[0]=0x01 indicates light switch-on, params[1-2]=0x0001 indicates delay time (unit: ms).
	Master sends read request to light module, and module will respond with data send_to_master[16]	See Packet sniffer log 2
OTA	OTA	Refer to OTA in Section 7
Pair	Pair function	

L2CAP Header		ATT_Write_Req								CRC
L2CAP-Length	ChanId	Opcode	AttHandle	AttValue						
0x0010	0x0004	0x12	0x0015	11 11 11	00 00	FF FF	D0 11 02	01 01 00		0x961183
		sno	src	dst	op	params				value[]

Figure 18 Packet sniffer log 1: Light switch-on command

Data Header				L2CAP Header		ATT_Read_Req		CRC	RSSI (dBm)	FCS
SN	MD	PDU-Length		L2CAP-Length	ChanId	Opcode	AttHandle			
0	0	7		0x0003	0x0004	0x0A	0x000D	0x0D8854	-38	OK

ATT_OP_READ_REQ=0x0A

Data Header				CRC	RSSI (dBm)	FCS
N	SN	MD	PDU-Length			
0	0	0		0xE78196	-38	OK

ATT_OP_READ_RSP=0x0B

Data Header				L2CAP Header		ATT_Read_Rsp														CRC	
SN	MD	PDU-Length		L2CAP-Length	ChanId	Opcode	AttValue														
1	0	21		0x0011	0x0004	0x0B	01 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00														0x23E5AB

send to master[16]

Figure 19 Packet sniffer log 2

6.8 Get online status

When “online status” function is used, if device address is empty during first power on, 8bit 0 and lower 8bit of 32-bit MAC address will be automatically configured as higher byte and lower byte of device address. If device address is overlapped for light nodes within the same network, it’s needed to manually configure the device address among the range of 0x0001~0x00FF, thus to report more user data.

Master can obtain the real-time information of lights in the mesh, including online/offline, on/off and brightness. Light module will notify the master with the status information change.

Mater can enable this function by writing ‘1’ to the character of “Status”.

L2CAP Header		ATT_Write_Req		
L2CAP-Length	ChanId	Opcode	AttHandle	AttValue
0x0004	0x0004	0x12	0x0012	01

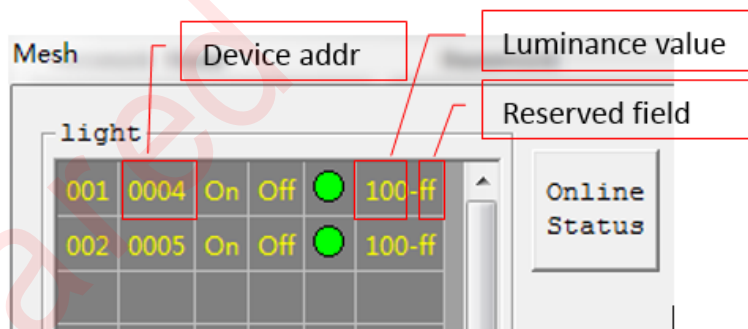
Figure 20 Enable “get online status”

L2CAP Header		ATT_Handle_Value_Notify										CRC	RSSI (dBm)	FCS
L2CAP-Length	ChanId	Opcode	AttHandle	AttValue										
0x0017	0x0004	0x1B	0x0012	00	00	00	00	00	00	00	00	DC 11 02	04	47 64 FF 00 00 00 00 00 00
												0x4A7A87	-38	OK

Figure 21 Online status data

Online status data format is as shown below:

- ✧ DC 11 02: Opcode field, which identifies the data as “online status” data.
- ✧ 04 47 64 FF: Data of one light node, including:
 - 04: It indicates device addr is 0x0004. Herein device address is not reported completely to enable more user data to be reported. To use “online status” function, the device address of the light node should be configured manually among the range of 0x0001~0x00FF.
 - 47: sno, the serial number for reported data.
 - 64: Light luminance value, ranging from 0 to 0x64.
 - FF: Field reserved for customer use.
- ✧ 00 00 00 00: Data of the second light. All “0” indicates empty data, i.e. this packet only contains data of one light.
- ✧ 00 00: Reserved field. Not available for user now.



6.9 Get light status

*Note: Master sends command similar to switch-on/off except for the command code definition varies. The light module will return data via notify.

After light's MAC_ID and MESH_INFO are configured, operation of "get light status" can be carried out with data structure defined as below:

```
rf_packet_att_cmd_t    pkt_light_status;
```

- 1) Use Master Dongle to read the light module's status. (Refer to guide in **Section 3**).

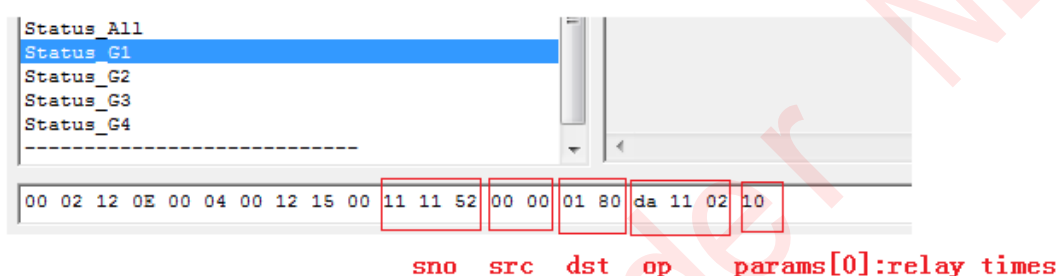


Figure 22 BLE light module status

- 2) Result captured via ble_sniffer tool:

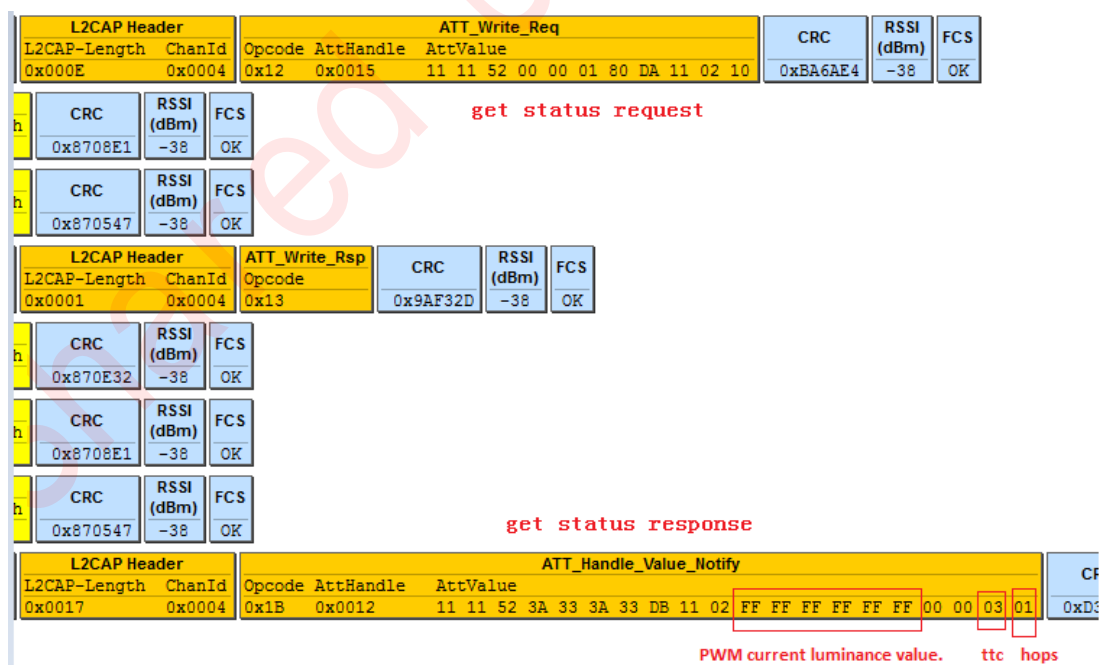


Figure 23 Packet sniffer result

*Note:

- 1) hops: This field serves to indicate hops needed for one command to be received

by certain light module.

Take master app->light1->light2->light3 for example, hops for light1, light2 and light 3 are 0, 1 and 2, respectively.

2) ttc: Higher two bits define accuracy: 0=ms,1=4ms,2=16ms,3=256ms. Lower six bits indicate the value.

For example, 0x42 (1000010) indicates accuracy is 4ms and value is 2. Delay time (Time to cost): 4ms*2=8ms.

6.10 Light initiatively sending notify data

By invoking the interface function “light_notify(u8 *p, u8 len)” in SDK, the light side at BLE connection state can initiatively send customized notify data (up to 10 data supported currently) to Master. The notify channel herein is the same as the channel used by “Get online status”, i.e. 0x1911.

L2CAP Header		ATT_Handle_Value_Notify										CRC	RSSI (dBm)	FCS
L2CAP-Length	ChanId	Opcode	AttHandle	AttValue										
0x0017	0x0004	0x1B	0x0012	00	00	00	00	00	00	00	00	EA 11 02 06	0x42D1E8	-38 OK

Figure 24 User notify data

User notify data format is as shown below:

- ✧ EA 11 02: Opcode field, which identifies the data as “user notify status” data.
- ✧ 06: Currently used for accumulation counting. Customizable.
- ✧ 00 00 00 00 00 00 00 00 00 00: User customizable data, default value is 0.

6.11 Scene mode

Up to 16 scene modes with configurable brightness and RGB values can be added for each light node within the mesh network.

First click the “Online Status” button (as shown in mark 1 of Figure 25).

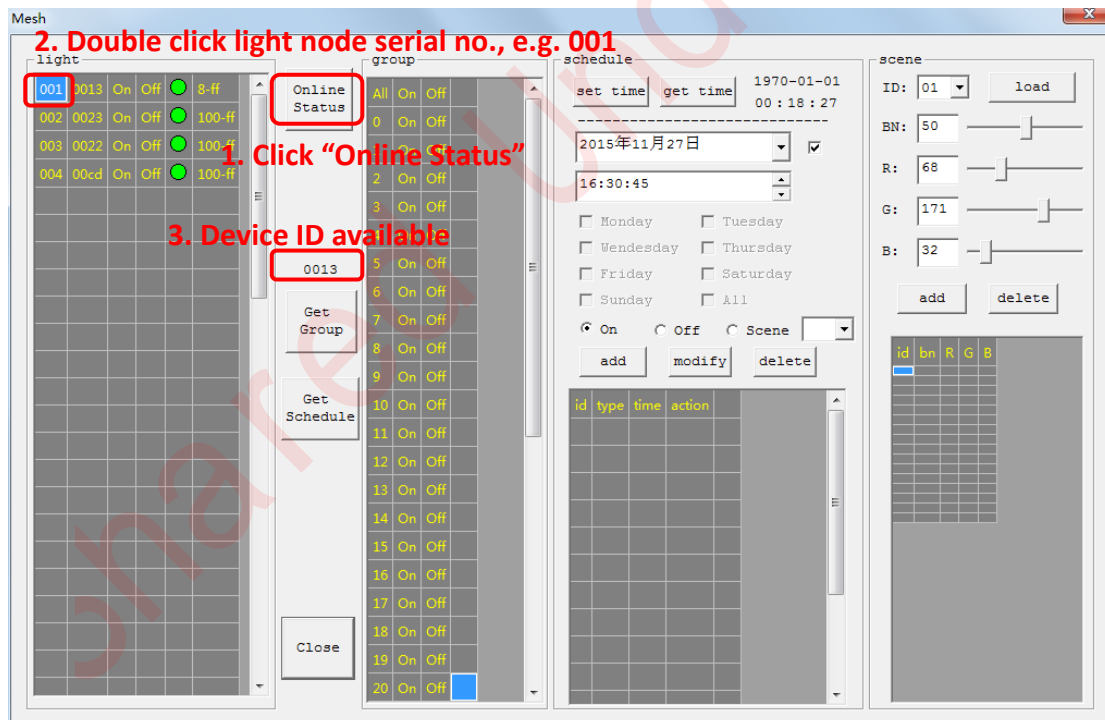
6.11.1 Add/Modify scene mode

The procedure to add scene mode is shown as below:

- 1) Double click certain light node’s serial number (e.g. 001 as shown in mark 2 of Figure 25); the node’s device address (ID) will be available above the “Get Group” button, as shown in mark 3 of Figure 25. *Note: If the addr shows “ffff”, the

operation is invalid.

- 2) Select scene ID in the drop-down menu of "ID", e.g. 01 as shown in mark 4 of Figure 25.
- 3) Set brightness and R/G/B values by inputting values into corresponding "BN", "R", "G" and "B" boxes or sliding corresponding buttons, as shown in mark 5 of Figure 25.
- 4) Click the "add" button (as shown in mark 6 of Figure 25) on the scene window. The indicating LEDs of the light node will blink slowly for three times. Then the scene mode 01 with specified brightness and RGB values is added for light node 001.
- 5) All scene information will be read by APP and updated at the bottom of scene window, as shown in mark 7 of Figure 25. "id" indicates scene ID; "bn"/"R"/"G"/"B" indicate brightness and RGB values, respectively.



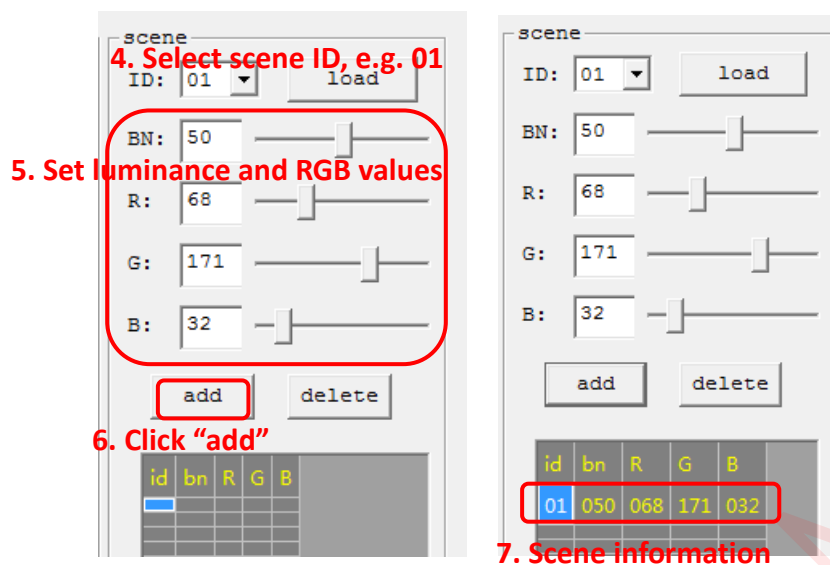


Figure 25 Add scene mode 01 for Light node 001 (device ID: 0013)

More than one scene mode (up to 16) can be added for one light node, as shown in Figure 26.

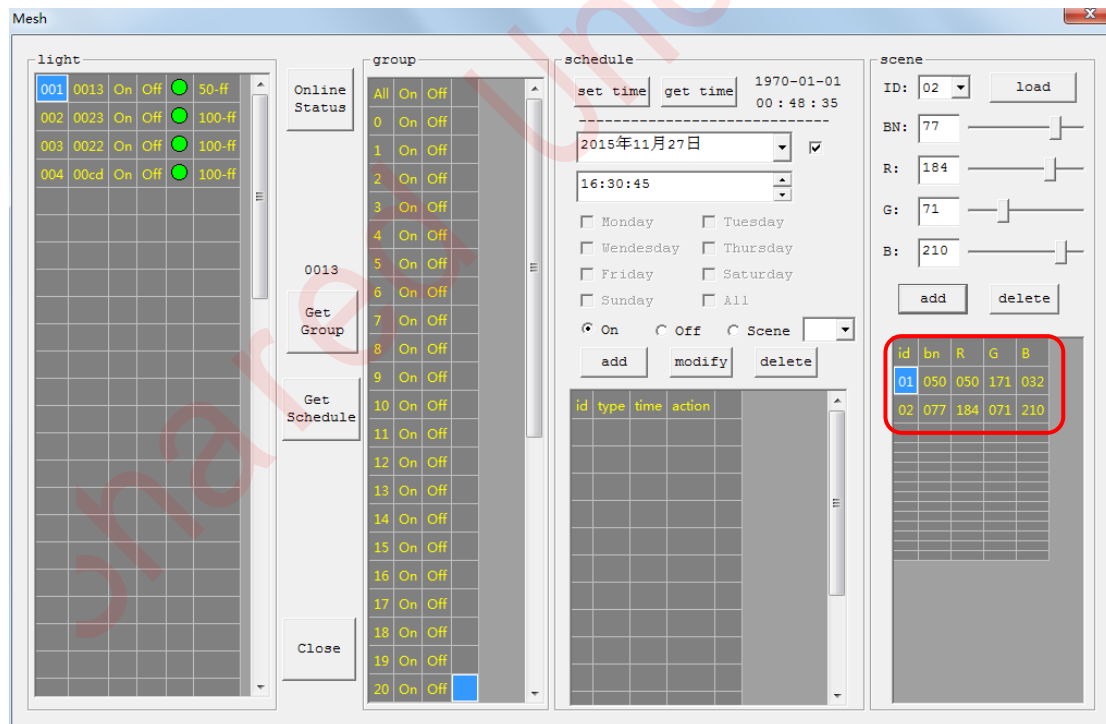


Figure 26 Add scene mode 01 and 02 to light node 001

When it's needed to modify brightness and RGB values for certain scene mode, first click the "id" number of the scene mode (as shown in mark 1 of Figure 27), then

modify BN/R/G/B values (as shown in mark 2 of Figure 27), finally click the “add” button again (as shown in mark 3 of Figure 27).

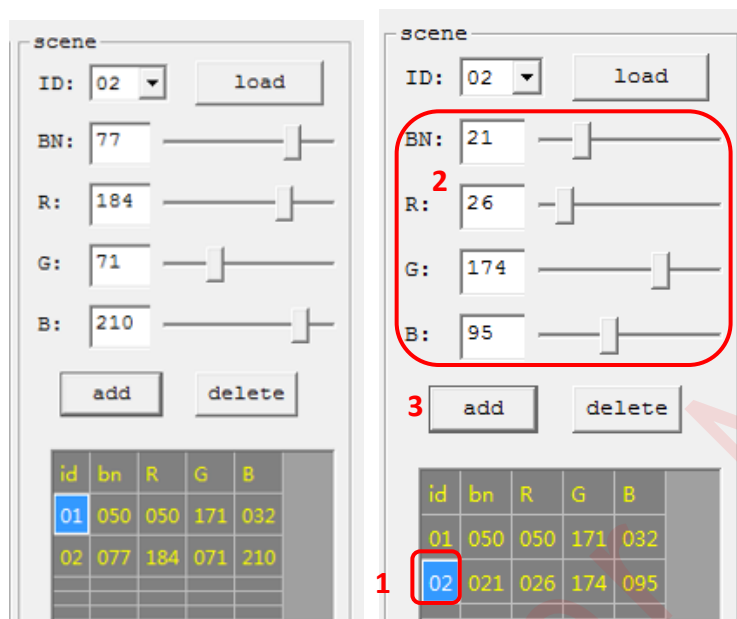


Figure 27 Modify scene mode 02

6.11.2 Load scene mode

After certain scene mode is added for one light node, it's needed to carry out scene mode loading steps to make it take effect.

- 1) Double click the light node's serial number. To load scene mode for current light node, this step can be skipped.
- 2) Select the scene ID to be loaded in the drop-down menu of “ID” (as shown in mark 1 of Figure 28), or directly click the corresponding “id” number (as shown in mark 2 of Figure 28).
- 3) Click the “load” button (as shown in mark 3 of Figure 28) on the scene window, the selected scene mode will be loaded to the corresponding light node.

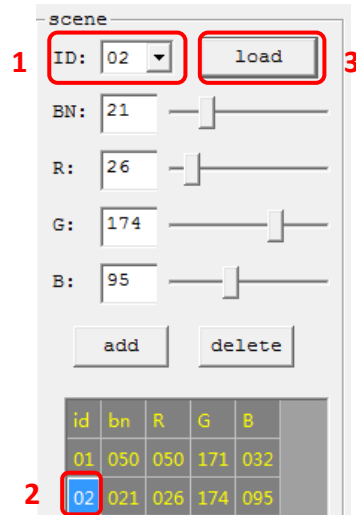
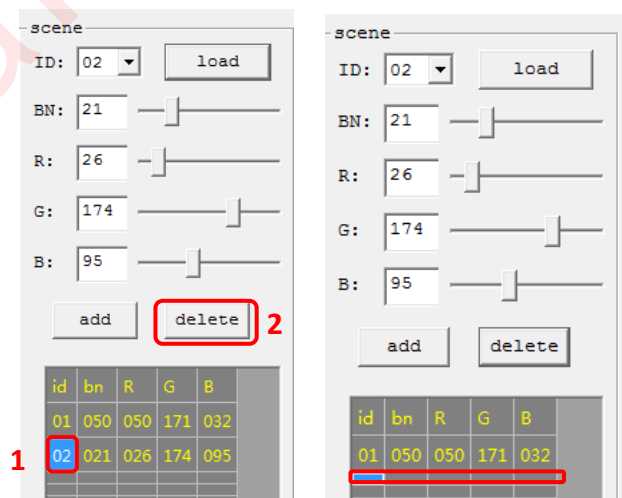


Figure 28 Load scene mode 02

6.11.3 Delete scene mode

The procedure to delete scene mode is shown as below:

- 1) Double click the light node's serial number. To delete scene mode for current light node, this step can be skipped.
- 2) Click the "id" number (as shown in mark 1 of Figure 29) corresponding to the scene mode to be deleted.
- 3) Click the "delete" button (as shown in mark 2 of Figure 29) on the scene window.
- 4) The indicating lights of the light node will blink slowly for three times; the selected scene mode is deleted, all scene information will be read by APP and updated at the bottom of scene window, as shown in mark 3 of Figure 29.



3. Information of scene mode 02 deleted

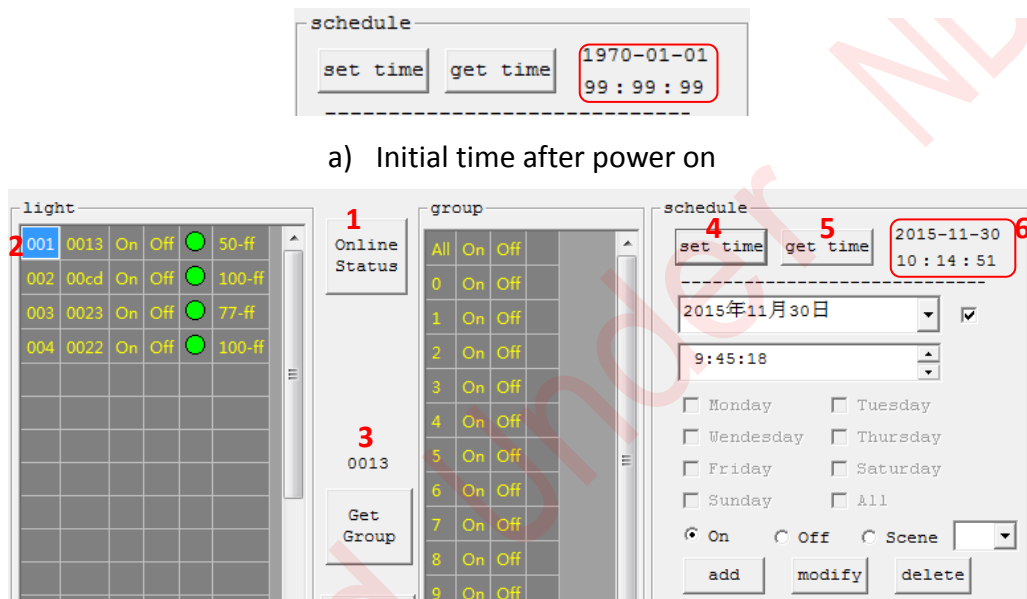
Figure 29 Delete scene mode 02

6.12 Schedule

6.12.1 Clock

The Clock function including “set time” and “get time” is supported for the light module.

As the light module does not embed RTC hardware module, the light’s time will be reset to “January 1st, 1970” after restart. So user needs to re-configure the time after powering on the light again.



b) Synchronize all nodes’ time to system time

Figure 30 Clock function

The procedure to set the time is shown as below:

- 1) Click the “Online Status” button (as shown in mark 1 of Figure 30).
- 2) Double click certain light node’s serial number (e.g. 001 as shown in mark 2 of Figure 30); the node’s device address (ID) will be available above the “Get Group” button, as shown in mark 3 of Figure 30. *Note: If the addr shows “ffff”, the operation is invalid.
- 3) Click the “set time” button (as shown in mark 4 of Figure 30), time of all nodes within the network will be synchronized to system time.
- 4) Light node’s time information will be available on the right side of the “get time” button, as shown in mark 6 of Figure 30.

- 5) By default the “Set time” command will be responded by all lights within the Mesh network. After the light receives and checks the parameters, its indicating light will blink slowly for three times to indicate successful setting; otherwise the indicating light will blink fast for three time to indicate the failure.

Clicking the “get time” button (as shown in mark 5 of Figure 30) is to manually obtain current time information of the light node.

***Note:** To avoid data loss due to collision in the air, the APP should get single light module’s time information at a time.

6.12.2 Alarm clock

The APP supports alarm clock information setting function, including “Add alarm clock”, “Obtain alarm clock information”, “Turn on/off alarm clock”, “Modify alarm clock” and “Delete alarm clock”.

First click the “Online Status” button, the device list will be available on the left of the window.

- 1) The procedure to **add an alarm clock** for certain light node is shown as below:
 - a) Double click one light’s serial number, e.g. 001 as shown in mark 1 of Figure 31. The light’s ID (device addr) will be available on the window, as shown in mark 2 of Figure 31.
 - b) Set alarm clock type: Currently two types including “DAY alarm clock” and “WEEK alarm clock” are defined.

The “DAY alarm clock” specifies the timing information in the form of “Date (i.e. month, day. The “year” is not supported now and the “DAY” type is defined as repeatable alarm clock with the period of one year.)” + “time (i.e. hour, minute, second)”; while the “WEEK alarm clock” specifies the timing information in the form of “Week (i.e. Monday,, Sunday. The “WEEK” type is also a repeatable alarm clock.)” + “time (i.e. hour, minute, second)”. When the alarm clock expires, the light module will respond with the pre-set event (“On” or “Off” or “Scene”).

The box as shown in mark 3 of Figure 31 is to select alarm clock type: ticking the box is to set the alarm clock as “DAY” type, otherwise the alarm clock is set as “WEEK” type.

- c) Set timing information: For “DAY” type, first set the “Date” information as shown in mark 4 of Figure 31; while for “WEEK” type, first set the “Week” information as shown in mark 5 of Figure 31. Then set the “time” information as shown in mark 6 of Figure 31.
- d) Set the light response event as shown in mark 7 of Figure 31: Ticking the “On”/“Off”/“Scene” circle indicates the light will be turned on/off or switch to the scene mode selected via the drop-down menu as a response when the alarm clock expires.
- e) Click the “add” button (as shown in mark 8 of Figure 31) to add the alarm clock. The APP will send a “read” command to read the alarm clock information. Mark 9 of Figure 31 shows the added alarm clock list: the “id”/“type”/“time”/“action” column indicates the index number, type, timing information and response event type for all of the light’s added alarm clocks.
- f) After the light module receives the command and checks the parameters, its indicating light will blink slowly for three times to indicate successful setting. Otherwise the indicating light will blink fast for three times to indicate the failure.

***Note:** For “WEEK” type alarm clock, the “week” information is indicated by bit[6:0] of one byte: bit[0:6] indicates “Sunday”, “Monday”, “Tuesday”.....“Saturday” respectively. Setting one bit to “1” is to select the corresponding day.

Take the alarm clock with the id number of “3” in the list as an example, the “week” information “0111110” indicate the alarm clock will respond from Monday to Friday.



Figure 31 Add alarm clock

- 2) The procedure to **obtain certain light's alarm clock information** is shown as below:
 - a) Double click one light's serial number, e.g. 002 as shown in mark 1 of Figure 32. The light's ID (device addr) will be available on the window, as shown in mark 2 of Figure 32.
 - b) The light's alarm clock information will be obtained automatically by the APP. Mark 3 of Figure 32 shows the obtained alarm clock information.
 - c) Clicking the "Get Schedule" button (as shown in mark 4 of Figure 32) is to obtain the light's alarm clock information manually.

***Note:** To avoid alarm clock data loss due to collision in the air, the APP should read single light's alarm clock information at a time.

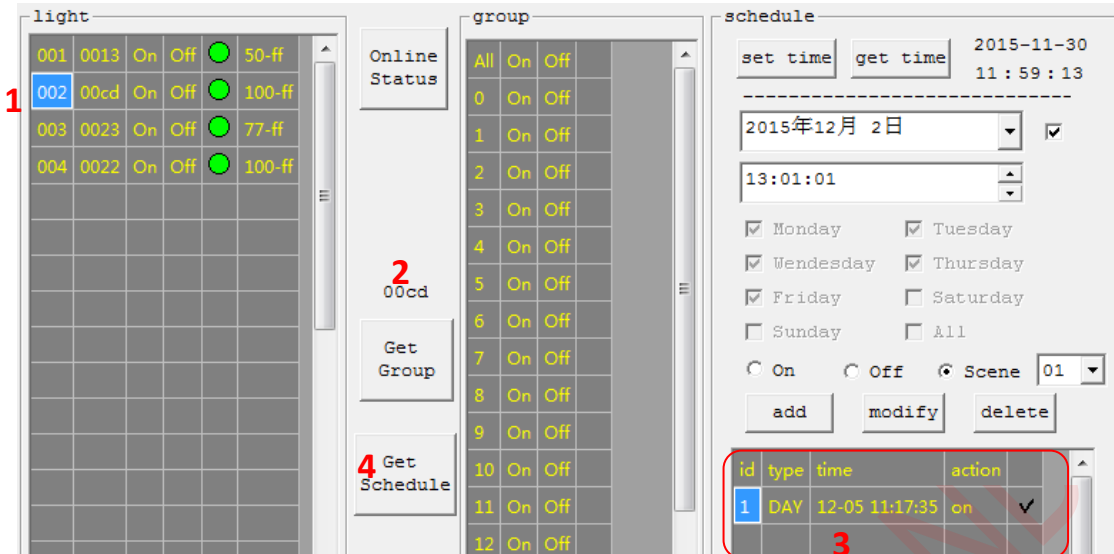


Figure 32 Obtain light's alarm clock information

- 3) The procedure to **turn on/turn off an alarm clock** is shown as below:
- a) Each added alarm clock status will be available on the last column of the list window. If one box is ticked, it indicates the corresponding alarm clock is turned on; otherwise it indicates the alarm clock is turned off.

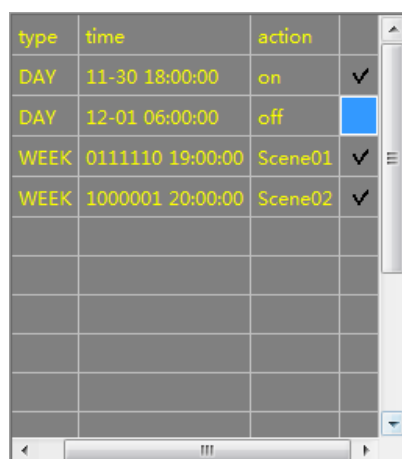
type	time	action	
DAY	11-30 18:00:00	on	✓
DAY	12-01 06:00:00	off	✓
WEEK	0111110 19:00:00	Scene01	✓
WEEK	1000001 20:00:00	Scene02	✓

All on by default

Figure 33 Alarm clock status

- b) Left/Right clicking the box of the last column is to turn on/turn off the corresponding alarm clock respectively. The APP will send a “read” command to read the alarm clock information and update the list correspondingly.

- c) After the light module receives the command and checks the parameters, its indicating light will blink slowly for three times to indicate successful setting; otherwise the indicating light will blink fast for three times to indicate the failure.



type	time	action	
DAY	11-30 18:00:00	on	✓
DAY	12-01 06:00:00	off	<input checked="" type="checkbox"/>
WEEK	0111110 19:00:00	Scene01	✓
WEEK	1000001 20:00:00	Scene02	✓

Figure 34 Turn off alarm clock 2

- 4) The procedure to **modify an alarm clock** is shown as below:
- Click the id number (as shown in mark 1 of Figure 35) of the alarm clock to be modified in the list window, and modify the parameters as needed (as shown in mark 2 of Figure 35).
 - Then click the “modify” button (as shown in mark 3 of Figure 35).
 - The APP will send a “read” command to read the alarm clock information and update the list correspondingly.
 - After the light module receives the command and checks the parameters, its indicating light will blink slowly for three times to indicate successful setting; otherwise the indicating light will blink fast for three times to indicate the failure.

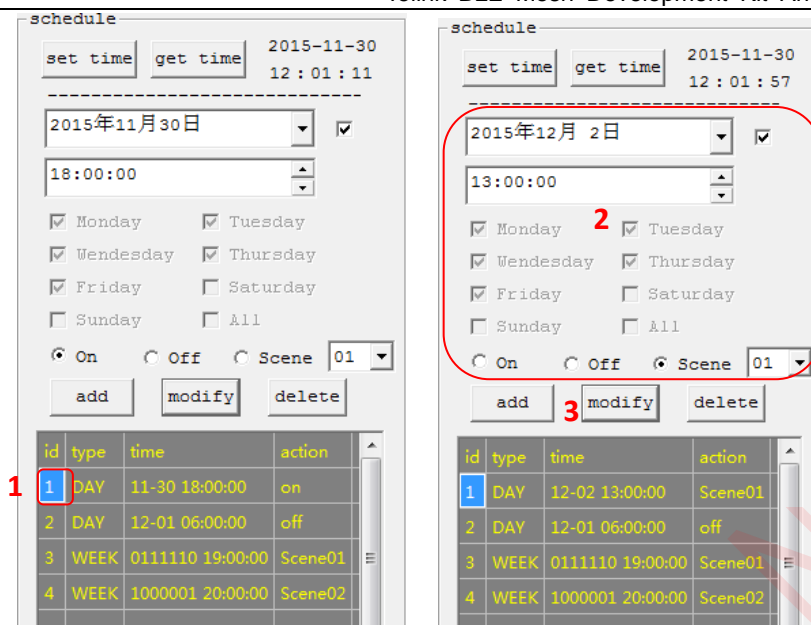


Figure 35 Modify alarm clock 1

5) The procedure to **delete an alarm clock** is shown as below:

- Click the id number (as shown in mark 1 of Figure 36) of the alarm clock to be deleted in the list window, and click the “delete” button (as shown in mark 2 of Figure 36).
- The APP will send a “read” command to read the alarm clock information and update the list correspondingly.
- After the light module receives the command and checks the parameters, its indicating light will blink slowly for three times to indicate successful setting; otherwise the indicating light will blink fast for three times to indicate the failure.

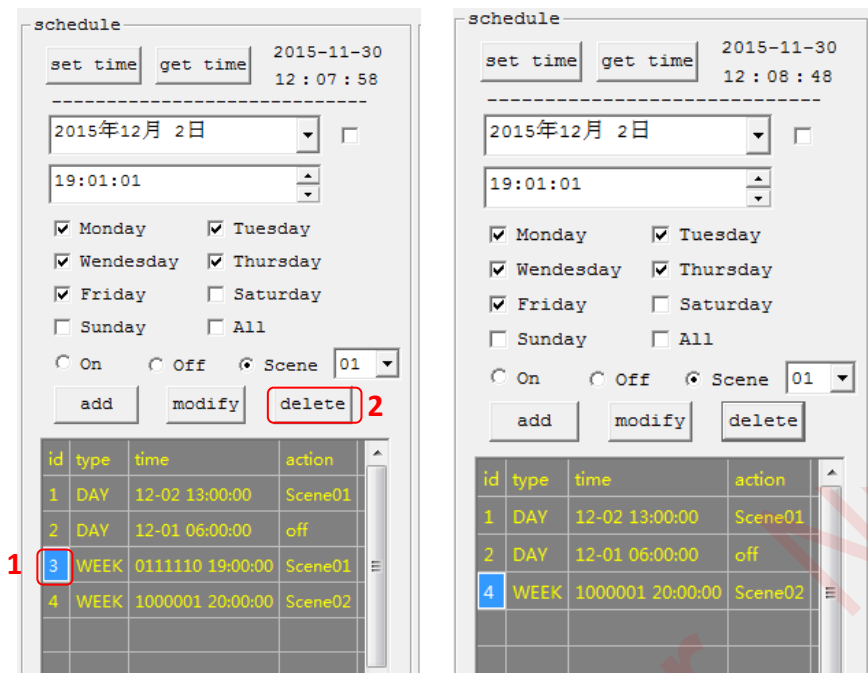


Figure 36 Delete alarm clock 3

6.13 Mesh

Prepare two light modules, assign them into the same mesh but different groups.

1) Setup:

Master->light1->light2, put the light2 far enough from Master so that only light1 can hear Master's command and light2 can only hear light1's transmission.

Master can't communicate directly with light2 due to distance limitation. If Master needs to send control command to light2, master first sends the command to light1, and then light1 relays the command to light2.

2) Add light module

If light3 needs to be added to the network, set its MESH_INFO same as the configuration of the mesh network, and use the command of "Add_Gxxx" to assign it to a group.

3) Delete light module

If light3 needs to be deleted from the network, just use the command of "Kick_out" to kick it out of the network. The light module will restore factory settings and its mesh name will change to the macro define of OUT_OF_MESH.

6.14 Flash storage areas

Table 10 Important flash storage address

No.	Flash address	Function
1	0x76000	MAC-ID, 6byte, unique factory value If not, randomly generated by system.
2	0x76010	Frequency offset calibration value. Negligible.
3	0x77000	Store pairing and group assignment information with 64byte as one unit.
		0~15 Byte[0]: Valid flag (0xFA)
		16~31 Mesh name
		32~47 Mesh PWD
		48~63 Long Term Key
4	0x78000	Store luminance information regulated by user (0-100%), RGB color index values and map_idx. Please refer to the data structure lum_save_t in SDK for details.
5	0x79000	Store device address and group information.
6	0x7A000	Power on counts for restore factory settings
7	0x7B000	Store alarm clock information.
8	0x7C000	Store scene information
9	0x7D000~0x7FFFF	Reserved or Customizable

7 OTA

Telink BLE light module supports firmware update OTA (Over the Air).

7.1 Use Master USB dongle as update tool

TBD.

7.2 Use APP as update tool

In this section, BLE-master-dongle is used to emulate the phone side.

- 1) Prepare one update board (i.e. Master USB dongle) and one light module.
- 2) Connect the update board, EVK board and PC. According to steps in Figure 37, download the ble_master_dongle.bin into flash address starting from 0x00 of the update board, and burn the new firmware (e.g. light2.bin) into the flash address starting from 0x20000.

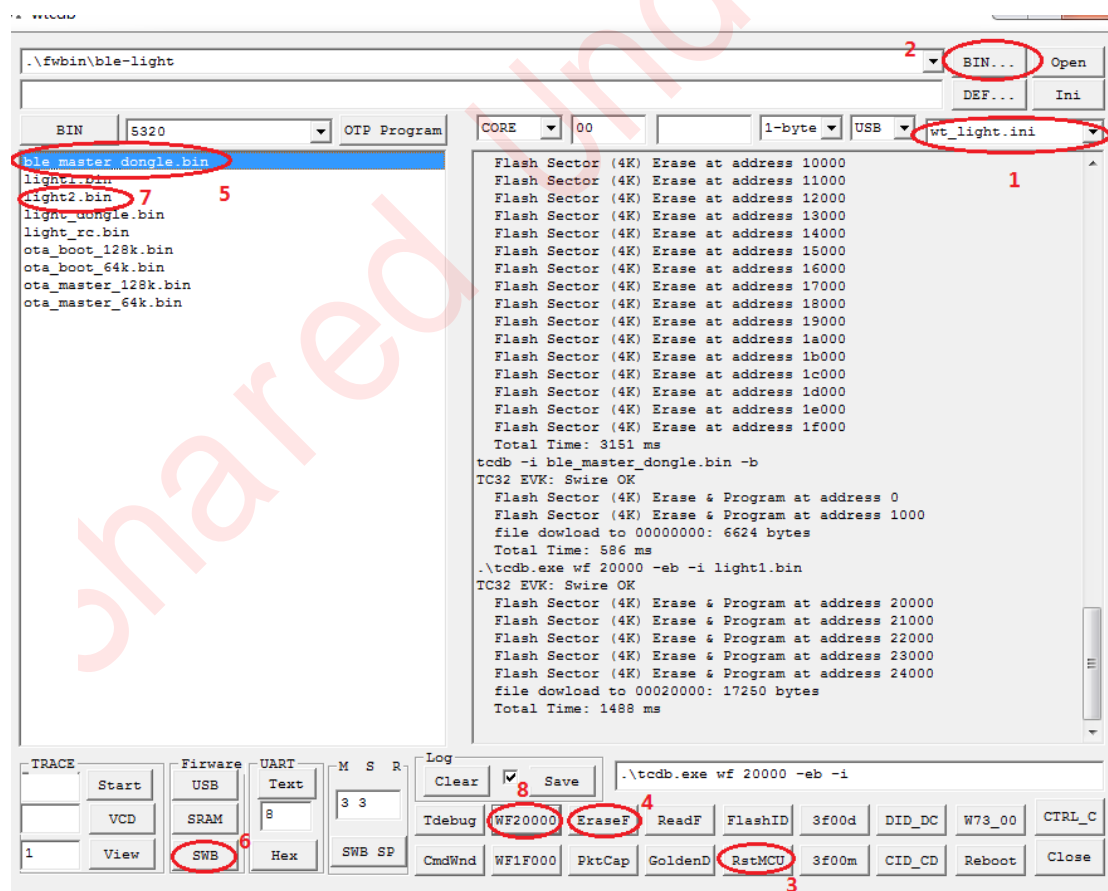


Figure 37 Burn ble_master_dongle.bin and new firmware into update board

- 3) Connect the light module, EVK and PC. According to steps in Figure 38, download the old firmware (e.g. light1.bin) into flash address starting from 0x00 of the light board, and burn the OTA boot loader (e.g. ota_boot_128k.bin) into the flash address starting from 0x1F000.

For the light module based on TLSR8267, this step can be skipped.

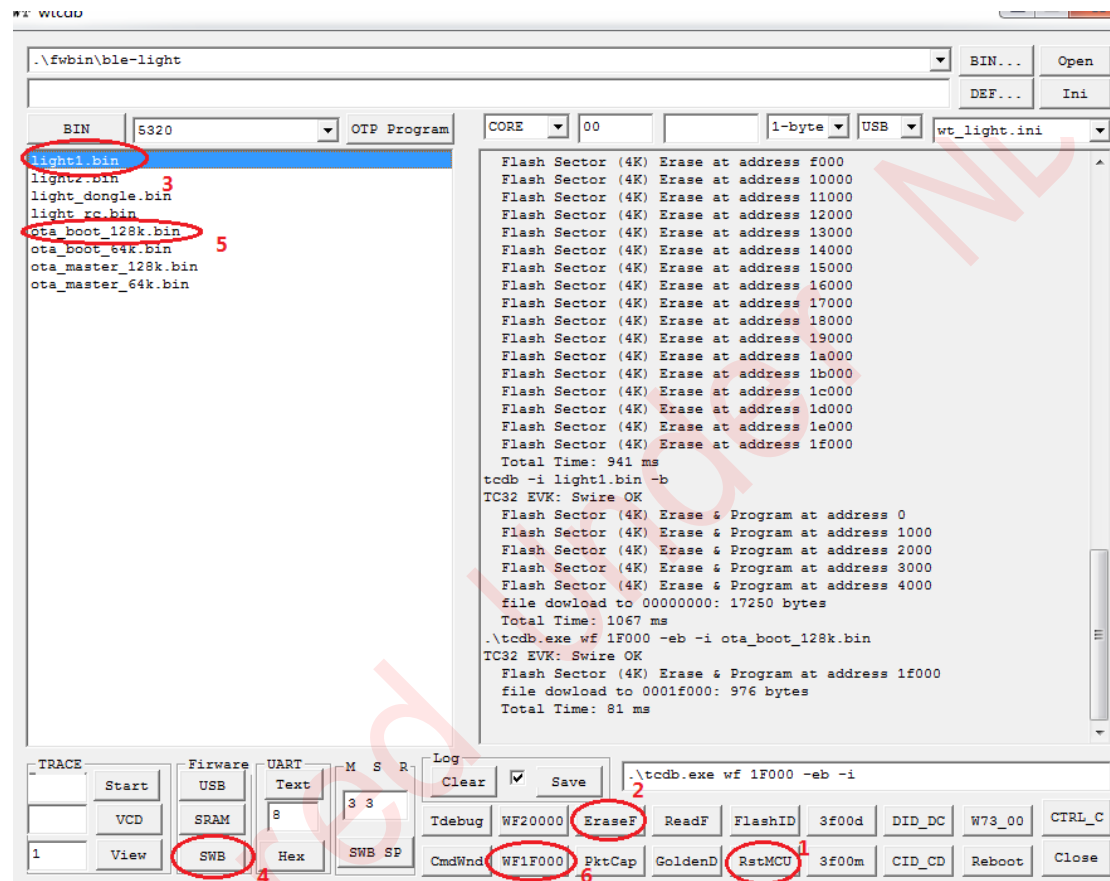


Figure 38 Burn old firmware and OTA boot loader into light board

4) OTA update flow:

- ✧ Open tl_ble_phone.exe, and follow steps 1~6 in Figure 39.

For non-encryption mesh, steps 4 and 5 can be skipped.

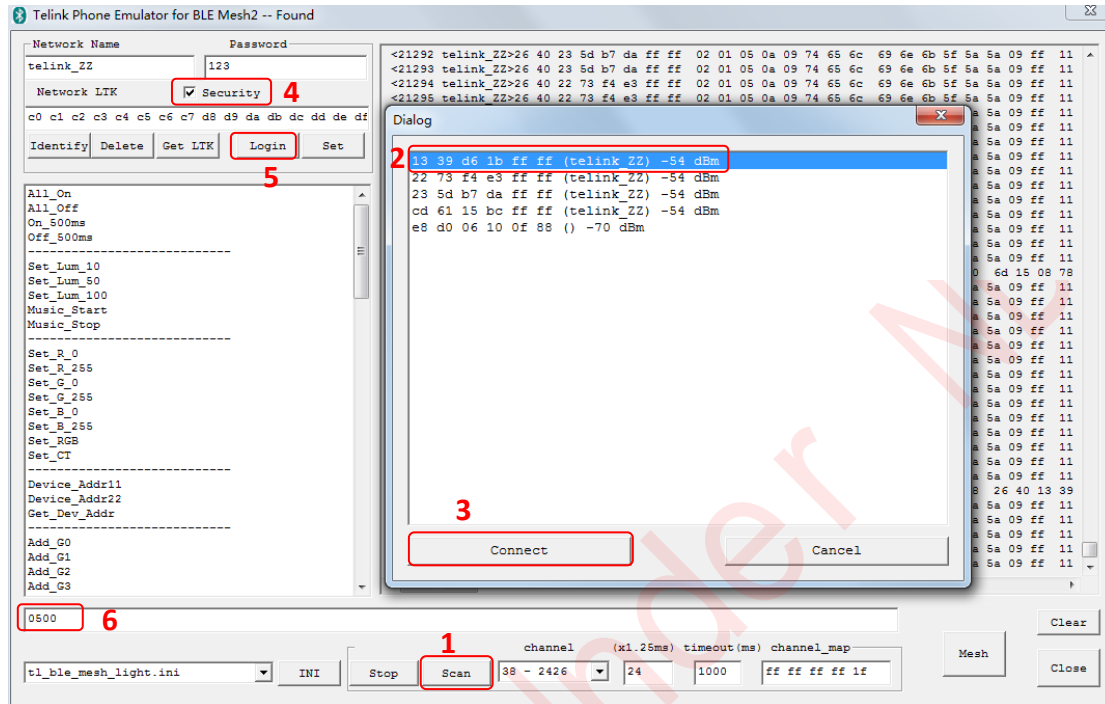


Figure 39 OTA update flow

*Note: Step 6 “0500+Enter (i.e. 0x0005)” indicates that OTA process starts.

- ✧ The light module will twinkle slowly for three times to indicate OTA update is successful (rf_led_ota_ok is invoked), while it will twinkle fast for three times to indicate that OTA process fails (rf_led_ota_error is invoked).
- ✧ Power down the light board and on again.

7.3 OTA development description on APP side



Figure 40 OTA update on APP side

- 1) The character of "OTA" is used for OTA update on APP side. APP sends data of light2.bin (new FW) to the character of "OTA" with the method of ATT_OP_WRITE_CMD(0x52). The data structure is defined in rf_packet_att_data_t with u8 dat[23] described as below:

Table 11 u8 dat[23]

dat[23]	Description
0~1	Serial number (starting from 0x0000)
2~17	16-byte data of the new bin file
18~19	crc value of previous 18 bytes
20~22	reserved

- 2) After all data of the bin file are transmitted, one more packet of "index+crc" needs to be transmitted to indicate the end of file transmission.
- 3) The dongle OTA update source code is in vendor/ble_master/rc_master.c with the main function of proc_ota for reference.
- 4) OTA packet is shown as below:

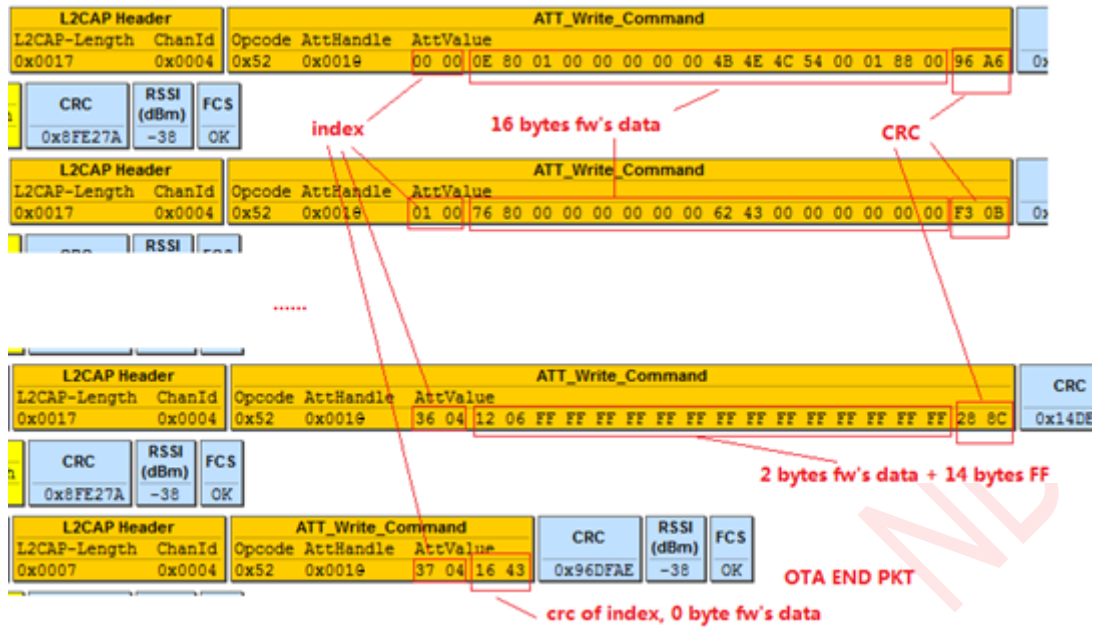


Figure 41 OTA packet

8 Factory Reset

Each light module can be restored to the default factory mode by the “Factory Reset” function which is manually triggered according to the following procedure:

- 1) Power on the Light module and down for three times with interval less than 3s.
The interval indicates the duration between each power on and power down.
- 2) Power on the Light module and down for two times with 3~30 seconds (3s and 30s not included) interval. The interval indicates the duration between each power on and power down.
- 3) Power on the Light module again. The red LED light on the module will blink slowly (0.5Hz frequency) for three times to indicate the Light module is successfully reset to factory defaults.

***Note:** The interval parameters are customizable via the following array:

```
u8 factory_reset_serials[SERIALS_CNT * 2] = { 0, 3,    // [0]:must 0
                                              0, 3,    // [2]:must 0
                                              0, 3,    // [4]:must 0
                                              3, 30,
                                              3, 30};
```

9 Firmware version management

Firmware version management file: `getver.sh`

Four bytes of last line indicates version number: `echo 0x01020304 >> $VER_FILE`

Before releasing new version firmware, the value will be updated, and stored in the 3rd - 6th byte of the firmware after clean compiling (clean old file information and re-compile).

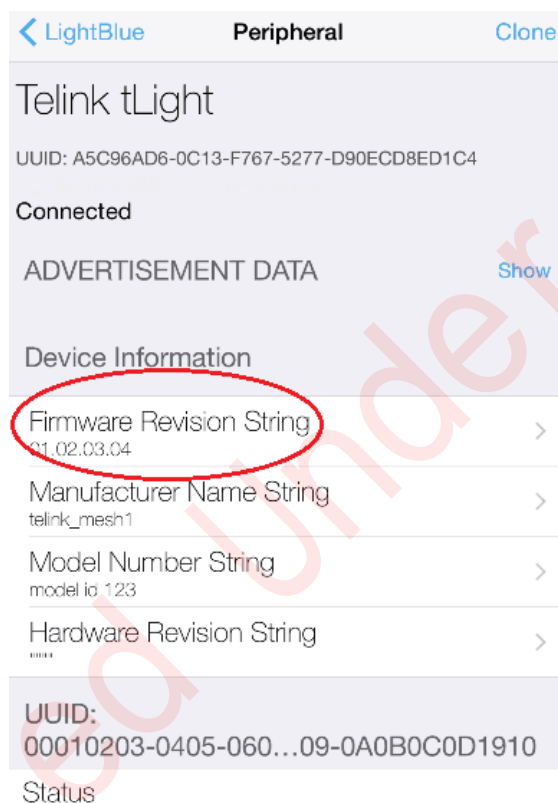


Figure 42 FW version information

10 Packet Sniffer

During connection/communication process between APP/Master dongle and light module, packet sniffer tool can be used to capture packet data. Burning Telink-provided "ble_sniffer_t.bin" into the dongle in the SDK will turn the dongle into a packet capture dongle.

The dongle can be plugged into the USB interface of PC for packet capture and it is compatible with both Wireshark and 3rd part packet analyzing tools (e.g. TI's packet sniffer tool).

11 Reference Documents

- ✧ Telink IDE User Guide
- ✧ Firmware burning and debugging User Guide
- ✧ Telink WtcdB User guide
- ✧ AN_BLE-15050400_User Guide For Telink BLE Mesh Emulator tool
- ✧ PS_BLE-15071300_Telink BLE Mesh Lighting APP Spec
- ✧ AN_BLE-16042500_Development Guide for BLE Mesh Lighting APP with Security
- ✧ AN_BLE-16021900_User Guide For Telink tLight APP