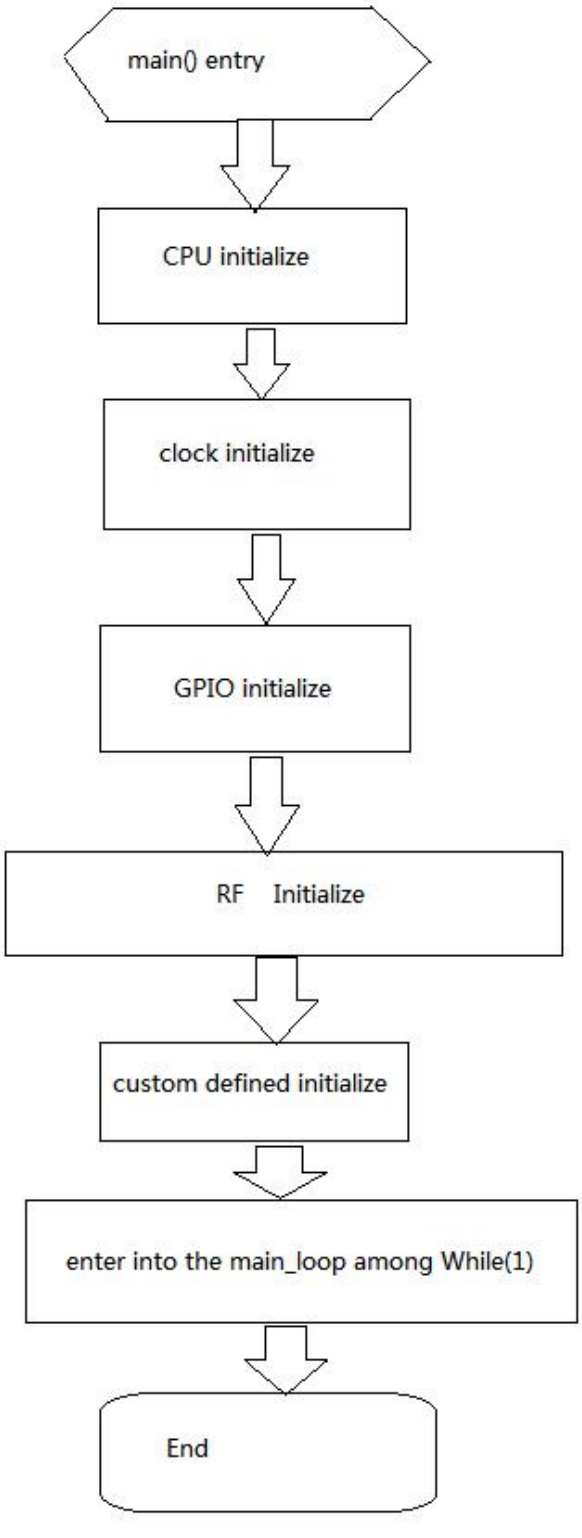
ST17H26 ble_sdk Dev Kit

1 Software Architecture



2 Main parameters

u8 tbl_mac [];

Introduction : used when ble_slave advertising MAC address(6 bytes)
e.g.： u8 tbl_mac [] = {0x11, 0x22, 0x33, 0x33, 0x22, 0x11};
(2) u8 tbl_adv [];
Description: Used to ble slave broadcast data. (where MAC address value is meaningless,will be modified by latter program)
e.g. :

```
u8 tbl_adv [] =
{0x0,   // reps adv type
0x1d,   // length = sizeof(tbl_adv – 5)
0xee, 0xe1, 0xe2, 0xe3, 0xe4, 0xe5, //mac address
0x05, 0x09, 'a', 'n', 't', 'i', //mac name
0x02, 0x01, 0x05, //Mark
0x03, 0x19, 0xC1, 0x03, //Appearance
0x07, 0x02, 0x03, 0x18, 0x03, 0x18, 0x0f, 0x18 //antiloss device
0,0,0   //CRC code which will be Complemented later among the program
};
```

(3) u8 tbl_rsp [];
Description:Used to be the scan response ble slave
E.g.: u8 tbl_rsp [] =

```
{0x04,   //reps   scan response
0x0f,   //length = sizeof(tb1_rsp-5)
0xef, 0xe1, 0xe2, 0xe3, 0xe4, 0xe5, //mac address
0x08, 0x09, 't', 'l','_', 'a', 'n', 't', 'i' //scan name "tl_anti"
0,0,0 //CRC code which will be Complemented later among the program
};
```

4) const attribute_t my_Attributes[];
This variable is initialized to the BLE protocol. These include the various sevice and the details of the character.Refer to <LENZE ble Attribute Table Description> Document.
(1) void My_att_init ();
Description: Used to initialize the BLE protocol stack.

3.1 Broadcast Ones

(1) blt_init (u8 *p_mac, u8 *p_adv, u8 *p_rsp);
Description: For ble slave end MAC address, broadcast data and scan response data initial Of。
Parameter 1:p_mac:: MAC address.
Parameter 2:P_ADV:: Broadcast data
Parameter 3:P_RSP:: Scan response data
Example: blt_init (tbl_mac, tbl_adv, tbl_rsp);
(2) void blt_set_adv_interval (u32 t_us);
Description: Sets the broadcast data interval.
Parameter 1:t_us:: Broadcast data interval unit US Example: blt_set_adv_interval (30000); Broadcast interval is 30ms
(3) void Rf_set_power_level_index (int level);

Note: Set the power of ble slave data. Parameter 1:level:: Contract power. The level values are selected as follows:

enum {
RF_POWER_8dBm = 0,
RF_POWER_4dBm = 1,
RF_POWER_0dBm = 2,
RF_POWER_m4dBm = 3,
RF_POWER_m10dBm = 4,
RF_POWER_m14dBm = 5,
RF_POWER_m20dBm = 6,
RF_POWER_m24dBm = 8,
RF_POWER_m28dBm = 9,
RF_POWER_m30dBm = 10,
RF_POWER_m37dBm = 11,
RF_POWER_OFF = 16,
};

Example: rf_set_power_level_index (RF_POWER_8DBM);
Set the ble slave end contract power to 8DBM.

(4) blt_send_adv (int mask); Note: Send broadcast data according to the parameter mask (broadcast data channel).

Normally, broadcasting Data channel for 37,38,39.

And in debugging, you can use 1 channel, easy to grasp packet analysis. Parameter 1:mask:: Broadcast data channel value.

Mask values are selected as follows:

#define blt_ENABLE_ADV_37 BIT (0)
#define blt_ENABLE_ADV_38 BIT (1)
#define blt_ENABLE_ADV_39 BIT (2)
#define blt_enable_adv_all (blt_enable_adv_37 |
blt_enable_adv_38 |
blt_ENABLE_ADV_39)

Example: blt_send_adv (blt_enable_adv_38)//only broadcast on channel
blt_SEND_ADV (blt_enable_adv_39)//Only broadcast on the channel
blt_SEND_ADV (blt_enable_adv_all)//broadcast on 37/38/39 channel

3.2 Connection Parts

(1) blt_brx ();

Description: This function is used to process ble slave and ble master each time they send and receive packets.

So if you want the Blue tooth functions then you have to call this function.

(2) Update connection parameter interface

Description: In general,when ble slave need update the BLE connection parameters, then it has to Send to the master a “update parameter request”, BLE master selects the appropriate connection's parameters based on the value from the slave-side request .

And before confirm back the response package, Master will check whether the slave

parameter is appropriate or not and determine to accept or reject the parameter update.
In the Lenze BLE SDK, the parameter update procedure is

(1) Call the following function a) ;
(2) Call the following function b).

a) void blt_conn_para_request (u16 Min_interval, u16 Max_interval, u16 Latency, u16 timeout)

Note: This function is used before the ble slave needs to update the parameters for the current connection parameter.
But this function does not contain the send update parameter command.
The connection interval between the Slave and master updates is between Min interval between Max interval.
Parameter 1:min interval:: Indicates ble slave need to update parameters (connect the communication room
The minimum threshold value.
Parameter 2:max interval:: Indicates ble slave need to update the maximum threshold for the parameter.
Parameter 3:latency:: Indicates ble slave need to update parameters latency
Parameter 4:timeout:: Indicates how long the slave is not connected to the master to indicate a port disconnect Connection。

b) void blt_update_parameter_request ();

Description: This function is used for ble slave to send update parameter commands to ble master.

3.3 Other General Functions

(1) u8 blt_push_notify (u16 handle, u32 val, int len);
Description: ble slave end-to-end ble master notify data.
This function is used when Data size in general is less than or equal to 4bytes ,
Parameter 1:handle:: ble slave sends data to master via a handle.
Its Handle the settings of the reference variable my_attributes.
Parameter 2:val:: Sending data values
Parameter 3:len:: Size of Val value
Example: blt_push_notify (15,3,1)//notify value 3 of HANDLE:15 to the master
(2) u8 blt_push_notify_data (u16 handle, u8 *p, int len); Description: ble slave end-to-end ble master notify data.
Typically, when data is less than 4bytes, the data is sent using this function. Parameter 1:handle:: ble slave end sends data to master via a handle.
It Handles the settings of the reference variable my_attributes.
Parameter 2:p:: A pointer to send data.
Parameter 3:len:: Size of the data to be sent
Example：
u8 Buffer[8];
blt_push_notify_data (15,buffer,8)/ notify 8 bytes of HANDLE:15 to the master, these data is stored Where the pointer begins at the beginning of the ‘buffer’.
(3) u8 blt_enable_suspend (u8 en);

Note: Set the suspend state of the BLE connection process. Parameter 1:en:: This parameter is to enabling the suspend State when device is broadcasting or the during the connected process.
The EN value can be selected as follows:
#define SUSPEND_ADV BIT (0)
#define Suspend_conn BIT (1)
Example: blt_enable_suspend (SUSPEND_ADV);//This system will enter the suspend state only in broadcast state.
blt_enable_suspend (suspend_conn);//This system will enter the suspend state only in connection state.
blt_enable_suspend (Suspend_conn |SUSPEND_ADV)//State will enter the suspend state.whenn this chip is connected or in broadcast state ,
blt_enable_suspend (0);//This system will not enter in any state Suspend state.

3.4 Encryption Section

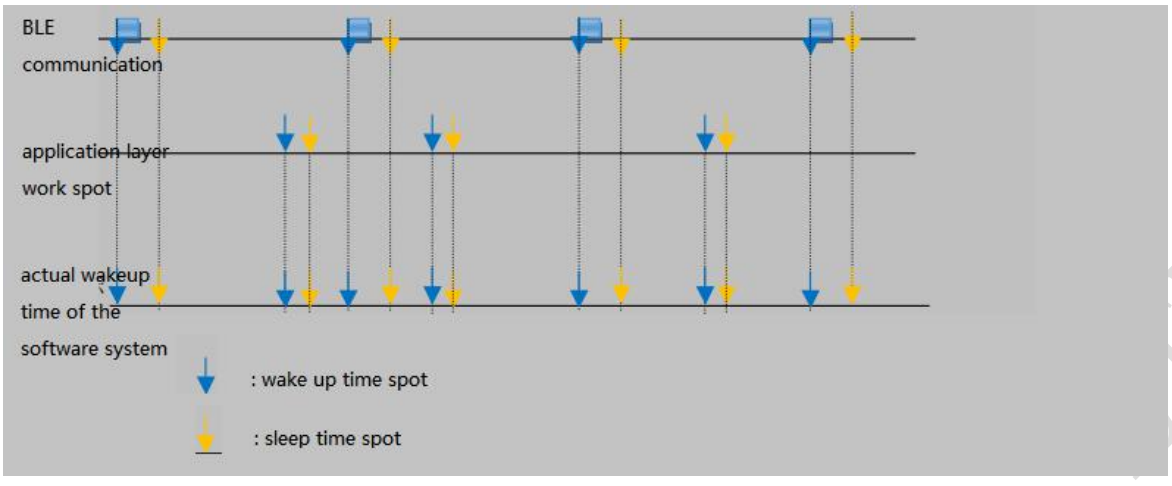
Description: In general encryption is required,when ble slave contains HID (Human Interface Device) service .
In addition, some mobile phones in original Bluetooth, will also require encryption (that is, BLE master will actively issue SMP paring Request).
(1) blt_smp_func_init ();
Note: This function needs to be called when the data needs encryption.
(1) blt_smp_store_in_ram_enable ();
Description: Data encryption information exists in RAM and will lose when power off.

3.5 callback function

Description: A series of callback functions are contained in LENZE ble SDK system , in order to set the flag for some important spots.
See "ble callback function explanation" for details

3.6 Sleep function

(1) void blt_brx_sleep (u32 app_wakeup_tick);
Description: This function is called to handle the idle state of the BLE connection process.
By default, when the next ble communication burst in,chip will quit this function.
Parameter 1:app_wakeup_tick:: The time that the user has customized to exit this function. This parameter is a time value, not a period value.



Enter Low power Consumption function:

int cpu_sleep_wakeup (int deepsleep, int wakeup_src, u32 wakeup_tick);

Description: This function is used to enter the system Low-power state and set the wake up source.

Once this function is called, the system goes into a low-power state.

Parameter: Deep Sleep: 0->reps : enter suspend mode; 1-> reps : enter Deep Sleep mode.

WAKEUP_SRC:: Indicates the wake up source.

The values can be selected as follows:

PM_WAKEUP_CORE ,//indicates the Digital sector is partially awakened.(e.g. Gpio) for suspend mode

PM_WAKEUP_TIMER ,//Indicates a timer wake-up, for suspend mode and Deep Sleep

PM_WAKEUP_PAD ,//For Gpio Awakening in Deep Sleep mode.

Wakeup_tick:: Time setting for timer wake-up,.The time is a future time spot not a period.

e.g.:

A. cpu_sleep_wakeup (0, pm_wakeup_timer, next_wakeup_tick)

After call the above function, the system will enter a suspend mode state.

You can wake up with a timer, wake up at the next_wakeup_tick.

Note: The next_wakeup_tick must be a future time (a spot time after the current system time).

B. cpu_sleep_wakeup (0, pm_wakeup_core, next_wakeup_tick)

After call the above function, the system enter the Suspend mode state. This state can only be awakened by changing the external IO state.

Note: Even if this The next_wakeup_tick value is not 0 and is not awakened by a timer.

If you use IO wakeup,you need to setup other configurations with IO ports

C. cpu_sleep_wakeup (1, pm_wakeup_timer, next_wakeup_tick)

After call the above function, the system enters the DeepSleep mode state.

This state can be awakened by a timer, with the suspend timer wakeup mechanism.

D. cpu_sleep_wakeup (1, pm_wakeup_pad, next_wakeup_tick)

After call the above function, the system enters the DeepSleep mode state. This state

can be only awakened by IO.
If you use IO wakeup, you need to setup other configurations with IO ports.
E. cpu_sleep_wakeup (1, pm_wakeup_pad |pm_wakeup_timer,next_wakeup_tick)
After call the above function, , the system enters a deepsleep mode state.which Can be awakened either by IO or by a timer.

(2) void blt_set_adv_type (u8 type);
Description: Sets the broadcast type. After setting, if in broadcast mode, it takes effect immediately.
Parameters: Type:: Broadcast types.
Broadcast type is one of the types:
EVENT_PKT_ADV = 0,//connected undirected advertising type
EVENT_PKT_CDA = 1,//connected directed advertising type
EVENT_PKT_NUA = 2,//non-connected undirected advertising
EVENT_PKT_DISCV = 6,//connected Discovery Advertising Type
Example:
blt_set_adv_type (EVENT_PKT_ADV);
When this function is called, the broadcast type is connected undirected advertising