# GL-Zigbee SDK User Guide

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Changed Item | Author | Date |
| 1.0 | First version | Feng.He | 2020.06.16 |

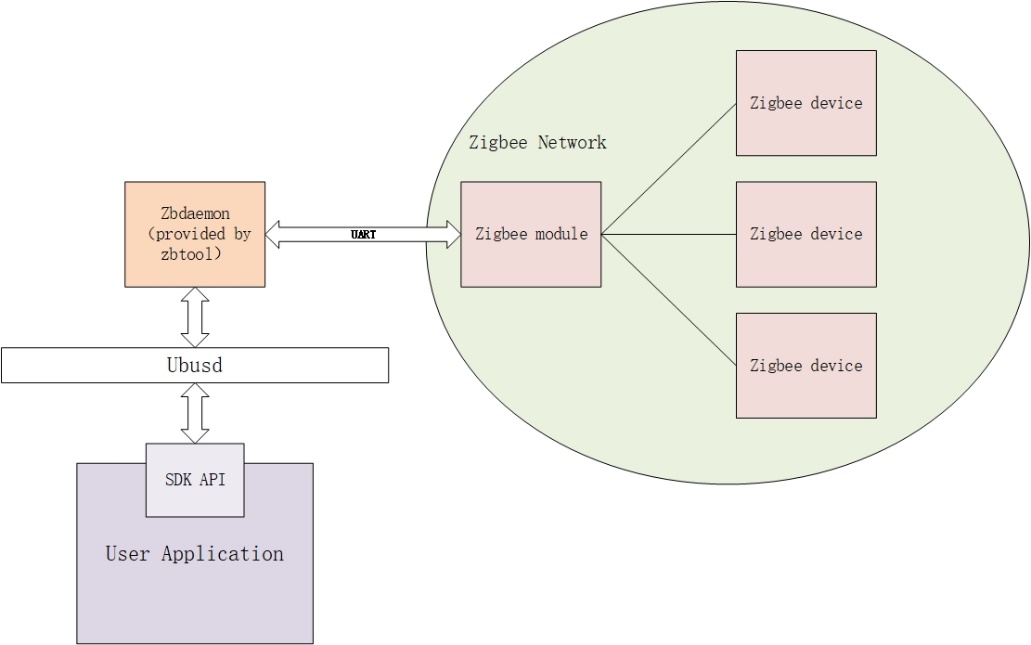
## 1. Description

### 1.1 What’s GL-Zigbee SDK

GL-Zigbee SDK is developed and provided by GL-iNet Technology as a part of the zigbee solution. The SDK encapsulates the communication between the gateway and the zigbee module, allowing third-party developers to focus on the business layer rather than the communication layer.

There are two parts of GL-Zigbee SDK, zbdaemon and gl-zb-api. Zbdaemon is used to communicate with modules and process zigbee protocol stack. Gl-zb-api encapsulates the communication between the user application and the zbdaemon. To maintain the stability of zbdaemon, we separated it from the SDK, created and managed it in zbtool. So developers only need to call the apis, no need to maintain zbdaemon.

The overall structure is as follows:



### 1.2 How to install zbtool

Because we haven't officially released the zbtool, you can only ssh to the router and install it using opkg.

opkg update

opkg install gl-zbtool\_1.0.2\_ipq806x.ipk

### 1.3 Directory structure of gl-zigbee\_sdk

|—— Makefile

|

|—— src

|—— doc # document

| |—— version\_log.md

|

|—— include # header file

| |—— libglzbapi.h

| |—— attribute-id.h

| |—— gl\_z3\_base.h

| |—— gl\_zb\_type.h

|

|—— lib # zb api lib

| |—— libglapi.a

|

|—— main # user application file

| |—— cli.c # demo file

|

|—— Makefile

## 2. API References

### 2.1 gl\_zb\_init()

int gl\_zb\_init(void);

Summary: Initialize the sdk.

Parameters: Void

Return Values:

|  |  |
| --- | --- |
| value | description |
| 0 | success |
| non-zero | fail |

### 2.2 gl\_zb\_free()

int gl\_zb\_free(void);

Summary: end of the program and release the resource.

Parameters: Void

Return Values:

|  |  |
| --- | --- |
| value | description |
| 0 | success |
| non-zero | fail |

### 2.3 gl\_zb\_subscribe()

int gl\_zb\_subscribe(void);

Summary: Enable callback functions(see in gl\_zb\_register\_cb()), subscribe module message.

Parameters: Void

Return Values:

|  |  |
| --- | --- |
| value | description |
| 0 | success |
| non-zero | fail |

### 2.4 gl\_zb\_unsubscribe()

int gl\_zb\_unsubscribe(void);

Summary: Disable callback functions(see in gl\_zb\_register\_cb()), unsubscribe module message.

Parameters: Void

Return Values:

|  |  |
| --- | --- |
| value | description |
| 0 | success |
| non-zero | fail |

### 2.5 gl\_zb\_register\_cb()

int gl\_zb\_register\_cb(gl\_zb\_cbs \*cb);

Summary: Register user callback functions

Parameters: gl\_zb\_cbs \*cb

typedef struct {

    int (\*z3\_zcl\_report\_cb)(gl\_zb\_zcl\_report\_p\* zcl\_p);

    int (\*z3\_dev\_manage\_cb)(gl\_z3\_desc\_s \*dev);

} gl\_zb\_cbs;

Return Values:

|  |  |
| --- | --- |
| value | description |
| 0 | success |
| non-zero | fail |

#### 2.5.1 int (\*z3\_zcl\_report\_cb)(gl\_zb\_zcl\_report\_p\* zcl\_p)

Summary: Receive zcl message reports from the module. User can get report message and use it in this callback. This callback will be called when module receive a zcl data(report or response).

Parameters: gl\_zb\_zcl\_report\_p\* zcl\_p

typedef struct {

    uint16\_t short\_id;

    uint16\_t profile\_id;

    uint16\_t cluster\_id;

    uint8\_t src\_endpoint;

    uint8\_t dst\_endpoint;

    uint8\_t cmd\_type;

    uint8\_t cmd\_id;

    uint16\_t msg\_length;

    uint8\_t \*message;

} gl\_zb\_zcl\_report\_p;

|  |  |
| --- | --- |
| parameter | description |
| short\_id | Short ID of report deivce |
| profile\_id | Profile ID of this message |
| cluster\_id | Cluster ID of this message |
| src\_endpoint | Source endpoint of this message |
| dst\_endpoint | Destination endpoint of this message |
| cmd\_type | 0: global zcl cmd; 1: specific zcl cmd |
| cmd\_id | ZCl command ID of this message |
| msg\_length | The length of data |
| message | Data |

Return Values:

|  |  |
| --- | --- |
| value | description |
| 0 | success |
| non-zero | fail |

#### 2.5.2 int (\*z3\_dev\_manage\_cb)(gl\_z3\_desc\_s \*dev)

Summary: Receive zigbee device message reports from the module. User can get report message and use it in this callback. This callback will be called when a zigbee device join, rejoin or left the network.

Parameters: gl\_z3\_desc\_s \*dev

typedef struct {

    char eui64[DEVICE\_MAC\_LEN+1];                              uint16\_t short\_id;

    uint16\_t parent\_node\_id;

    gl\_device\_update status;

    gl\_Join\_decision decision;

} gl\_z3\_desc\_s;

|  |  |
| --- | --- |
| parameter | description |
| eui64 | Eui64 of new device |
| short \_id | Short ID of new device |
| parent\_node\_id | Short ID of the parent of new device |
| status | Status of the Update Device message |
| decision | The decision made by the Trust Center when a node attempts to join. |

Return Values:

|  |  |
| --- | --- |
| value | description |
| 0 | success |
| non-zero | fail |

### 2.6 gl\_zb\_get\_module\_message ()

int gl\_zb\_get\_module\_message(gl\_zb\_module\_version\* status);

Summary: Get module software information.

Parameters: gl\_zb\_module\_version\* status

typedef struct {

    uint16\_t build;

    uint8\_t major;

    uint8\_t minor;

    uint8\_t patch;

    uint8\_t special;

    uint8\_t type;

} gl\_zb\_module\_version;

Return Values:

|  |  |
| --- | --- |
| value | description |
| 0 | success |
| non-zero | fail |

### 2.7 gl\_zb\_get\_nwk\_status()

int gl\_zb\_get\_nwk\_status(gl\_zb\_nwk\_status\_para\* status);

Summary: Get current network status.

Parameters: gl\_zb\_nwk\_status\_para\* status

typedef struct {

    gl\_zb\_nwk\_status nwk\_status;

    gl\_zb\_node\_type node\_type;

    char extended\_pan\_id[33];

    uint16\_t pan\_id;

    uint8\_t radio\_tx\_power;

    uint8\_t radio\_channel;

    gl\_join\_method join\_method;

    uint16\_t nwk\_manager\_id;

    uint8\_t nwk\_update\_id;

} gl\_zb\_nwk\_status\_para;

|  |  |
| --- | --- |
| parameter | description |
| nwk\_status | State of device |
| node\_type | Type of device in current network |
| extended\_pan\_id | Extended pan ID of current network |
| pan\_id | Pan ID of current network |
| radio\_tx\_power | Radio TX power |
| radio\_channel | Radio channel |
| join\_method | The type of method used for joining. |
| nwk\_manager\_id | The ID of the network manager in the current network. |
| nwk\_update\_id | The value of the ZigBee nwkUpdateId known by the stack. |

Return Values:

|  |  |
| --- | --- |
| value | description |
| 0 | success |
| non-zero | fail |

### 2.8 gl\_zb\_create\_nwk()

int gl\_zb\_create\_nwk(uint16\_t pan\_id, uint8\_t channel, uint8\_t tx\_power);

Summary: Create a new zigbee network(as coordinator).

Parameters:

|  |  |
| --- | --- |
| parameter | description |
| pan\_id | Pan ID of network |
| channel | Channel of network |
| tx\_power | Radio TX power |

Return Values:

|  |  |
| --- | --- |
| value | description |
| 0 | success |
| non-zero | fail |

### 2.9 gl\_zb\_leave\_nwk()

int gl\_zb\_leave\_nwk(void);

Summary: leave the network.

Parameters: Void

Return Values:

|  |  |
| --- | --- |
| value | description |
| 0 | success |
| non-zero | fail |

### 2.10 gl\_zb\_allow\_dev\_join()

int gl\_zb\_allow\_dev\_join(int limit\_time);

Summary: open the network, allow new device join in.

Parameters:

|  |  |
| --- | --- |
| parameter | description |
| limit\_time | The time of open network |

Return Values:

|  |  |
| --- | --- |
| value | description |
| 0 | success |
| non-zero | fail |

### 2.11 gl\_zb\_delete\_dev()

int gl\_zb\_delete\_dev(char\* mac, uint16\_t short\_id);

Summary: Remove device from zigbee network.

Parameters:

|  |  |
| --- | --- |
| parameter | description |
| mac | Eui64 of target device |
| short\_id | Short ID of target device |

Return Values:

|  |  |
| --- | --- |
| value | description |
| 0 | success |
| non-zero | fail |

### 2.12 gl\_zb\_get\_dev\_table()

int gl\_zb\_get\_dev\_table(gl\_zb\_dev\_table \*table);

Summary: Get current child/neighbor device table. A child device is usually a zigbee-end\_device or zigbee-sleepy\_end\_device mounted on the current device. A neighbor device is usually a zigbee-router\_device.

Note: if an end-device mounted on other router, it will not show in child table.

Parameters: gl\_zb\_dev\_table \*table

typedef struct child\_table\_node{

    char eui64[33];

    gl\_zb\_node\_type type;

    uint16\_t short\_id;

    uint8\_t phy;

    uint8\_t power;

    uint8\_t timeout;

    struct child\_table\_node\* next;

} gl\_child\_table\_node;

typedef struct neighbor\_table\_node{

    uint16\_t short\_id;

    uint8\_t average\_lqi;

    uint8\_t in\_cost;

    uint8\_t out\_cost;

    uint8\_t age;

    char eui64[33];

    struct neighbor\_table\_node\* next;

} gl\_neighbor\_table\_node;

typedef struct {

    int child\_num;

    gl\_child\_table\_node \*child\_table\_header;

    int neighbor\_num;

    gl\_neighbor\_table\_node \*neighbor\_table\_header;

} gl\_zb\_dev\_table;

Return Values:

|  |  |
| --- | --- |
| value | description |
| 0 | success |
| non-zero | fail |

### 2.13 gl\_zb\_send\_zcl\_cmd()

int gl\_zb\_send\_zcl\_cmd(gl\_zb\_zcl\_cmd\_p \*frame);

Summary: Create and send a zcl command.

Parameters: gl\_zb\_zcl\_cmd\_p \*frame

typedef struct {

    uint16\_t short\_id;

    uint16\_t profile\_id;

    uint16\_t cluster\_id;

    uint8\_t src\_endpoint;

    uint8\_t dst\_endpoint;

uint8\_t cmd\_type;

    uint8\_t cmd\_id;

uint8\_t frame\_type;

    uint16\_t msg\_length;

    uint8\_t \*message;

} gl\_zb\_zcl\_cmd\_p;

|  |  |
| --- | --- |
| parameter | description |
| short\_id | Short ID of report deivce |
| profile\_id | Profile ID of this message |
| cluster\_id | Cluster ID of this message |
| src\_endpoint | Source endpoint of this message |
| dst\_endpoint | Destination endpoint of this message |
| cmd\_type | 0: global zcl cmd; 1: specific zcl cmd |
| cmd\_id | ZCl command ID of this message |
| frame\_type | 0: unicast; 1: multicast; 2: broadcast |
| msg\_length | The length of data |
| message | Data |

Note: not support multicast and broadcast now.

Return Values:

|  |  |
| --- | --- |
| value | description |
| 0 | success |
| non-zero | fail |