**VIETTEL GROUP**

**VIETTEL HIGH TECHNOLOGY INDUSTRIES CORPORATION**



SDK Innoway User Guide

(FOR DEVICE USING ESP32-MICROCONTROLLER)

REVISION HISTORY

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Version** | **Changed contents** | **Date** | **Author** |
| 01 | V.1 | Create new | 24/02/2023 |  |

**CONTENTS**

[REVISION HISTORY 2](#_Toc128331220)

[Table of Figures 4](#_Toc128331221)

[List of tables 4](#_Toc128331222)

[I. Documentation Description 5](#_Toc128331223)

[II. Demostration project 5](#_Toc128331224)

[1. Project & Folder 5](#_Toc128331225)

[2. Configuration project 6](#_Toc128331226)

[3. Mobile App and Testing 6](#_Toc128331227)

[3.1. Demo device 6](#_Toc128331228)

[3.2. Mobile App 6](#_Toc128331229)

[III. Library description 7](#_Toc128331230)

[1. Device integrated to the Innoway Platform 7](#_Toc128331231)

[1.1. Description 7](#_Toc128331232)

[1.2. Quick pair mode 8](#_Toc128331233)

[1.3. Example test quick mode 8](#_Toc128331234)

[1.4. Compatible pair mode 9](#_Toc128331235)

[1.5. Example test compartible mode 10](#_Toc128331236)

[2. Connecting to MQTT Innoway 10](#_Toc128331237)

[2.1. Description 10](#_Toc128331238)

[2.2. Example test 10](#_Toc128331239)

[3. Connecting to NB-IoT 11](#_Toc128331240)

[3.1. Description 11](#_Toc128331241)

[3.2. Example test 11](#_Toc128331242)

[References 12](#_Toc128331243)

Table of Figures

[Figure 1. Demo project and folder 5](#_Toc129850495)

[Figure 2. Device configuration 6](#_Toc129850496)

[Figure 3. Device on the VHome App 7](#_Toc129850497)

[Figure 4. Logging device information 9](#_Toc129850498)

[*Figure 5. Pinout BC660K Module* 12](#_Toc129850499)

[*Figure 6. Logging in Terminal* 12](#_Toc129850500)

List of tables

[Table 1. Define Type of wi-fi device 8](#_Toc128331248)

1. Documentation Description

The demo project is created to illustrate how a device connect to Innoway Platform. This help 3­­­­­rd partners to develop their products which can be integrated with Innoway Platform.

The project is written in C language (the normal language used in device firmware). Most part are opened in source code format to help developers to understand faster about the ideas of the project. The libraries are built on ESP-IDF SDK v4.4 used for ESP32, ESP32-S2, ESP32-C3 microcontrollers.

1. Demostration project
   1. Project & Folder

Software used in sample projects:

* + Espressif IDE: Programming Tools
  + VHome Mobile App: Integrated application, device management.

Graphical user interface, application

Description automatically generated

Figure 1. Demo project and folder

The sample project integrates wi-fi devices onto the Innoway platform and uses the MQTT protocol to communicate with the platform.

Project folder description:

* mqtt\_innoway.h: Connect/disconnect MQTT, publish/subscribe topic.
* pair\_innoway.h: Integrate the device onto the Innoway platform.
* libmqtt\_innoway.a, libpair\_innoway.a: static libraries, provided in the our SDK
* CmakeLists.txt in main folder: Configuration for the project
  1. Configuration project

In the CmakeLists.txt the file in the main folder must include the following commands to add mqtt\_innoway.h and pair\_innoway.h:

add\_prebuilt\_library(libmqtt\_innoway "libmqtt\_innoway.a"

PRIV\_REQUIRES mqtt log)

target\_link\_libraries(${COMPONENT\_LIB} PRIVATE libmqtt\_innoway)

add\_prebuilt\_library(libpair\_innoway "libpair\_innoway.a"

REQUIRES esp\_eth esp\_event esp\_http\_server esp\_wifi log mqtt nvs\_flash wpa\_supplicant json)

target\_link\_libraries(${COMPONENT\_LIB} PRIVATE libpair\_innoway)

* 1. Mobile App and Testing
     1. Demo device

The information of the demo Gateway device:

Graphical user interface, text, application

Description automatically generated

Figure 2. Device configuration

* + 1. Mobile App

Result of a successful integration is that we can use the current mobile app “VHome” to view and control the 3rd partner’s devices.

3rd partner should use our “VHome” app for testing in integration step. There are both versions in IOS and Android. We will help 3rd partner to add the demo camera and an account to use VHome App.

Graphical user interface, application

Description automatically generated Graphical user interface, application

Description automatically generated

Figure 3. Device on the VHome App

1. Library description
   1. Device integrated to the Innoway Platform
      1. Description

The pair\_innoway library is used to integrate wi-fi devices onto the Innoway platform. In the pair processing stream, the device and the Innoway platform exchange some credentials with each other.

Struct “deviceInfo” defines the information data type of a device in pair\_innoway.h:

**typedef** **struct** deviceInfo{

**char** deviceType[65];

**char** deviceName[33];

**char** deviceID[65];

**char** deviceToken[65];

}deviceInfo;

* deviceType: Device type definition, type list defined in Table 1
* deviceName: The device name when displayed on the VHome App, the user sets it.
* deviceID: The ID of the device after integration into the Innoway platform, is the MQTT client ID.
* deviceToken: The token that manages the user's login session, which is the MQTT client password.

Note: It is necessary to declare and initialize a deviceInfo struct type variable in the main program.

|  |  |  |
| --- | --- | --- |
| **No** | **Type of device** | **Type (String)** |
| 1 | Switch | 18 |
| 2 | Smart Socket | 19 |

Table 1. Define Type of wi-fi device

In this library, provide two functions that use two different pair modes:

* + 1. Quick pair mode

Function esp\_err\_t **startQuickPair**(deviceInfo\* device);

Input: Struct deviceInfor device

Output: ESP\_OK: Pair Success, ESP\_ERROR: Failed pair

Note: deviceID and deviceToken used to connect MQTT Innoway.

* + 1. Example test quick mode

Source code in main.c file:

**#include** <stdio.h>

**#include** <string.h>

**#include** "esp\_wifi.h"

**#include** "nvs\_flash.h"

**#include** "esp\_event.h"

**#include** "esp\_system.h"

**#include** "esp\_log.h"

**#include** "pair\_innoway.h"

**static** **const** **char**\* TAG = "MAIN";

**void** **app\_main**(**void**)

{

ESP\_ERROR\_CHECK(**nvs\_flash\_init**());

// Initialize the underlying TCP/IP stack

ESP\_ERROR\_CHECK(**esp\_netif\_init**());

// Create default event loop

ESP\_ERROR\_CHECK(**esp\_event\_loop\_create\_default**());

//Creates default WIFI AP. In case of any init error this API aborts

**esp\_netif\_create\_default\_wifi\_ap**();

// Creates default WIFI STA. In case of any init error this API aborts

**esp\_netif\_create\_default\_wifi\_sta**();

// Declare variable configuration default wifi

wifi\_init\_config\_t cfg = WIFI\_INIT\_CONFIG\_DEFAULT();

// Initialize WiFi

ESP\_ERROR\_CHECK( **esp\_wifi\_init**(&cfg) );

ESP\_LOGI(TAG, "STARTING TEST PAIR INNOWAY LIB");

deviceInfo HubZigbee = {

.deviceType = "1",

.deviceName = "Hub Zigbee",

.deviceID = "",

.deviceToken = ""

};

startQuickPair(&HubZigbee);

ESP\_LOGI(TAG, "DEVICE TYPE: %s", HubZigbee.deviceType);

ESP\_LOGI(TAG, "DEVICE NAME: %s", HubZigbee.deviceName);

ESP\_LOGI(TAG, "DEVICE ID: %s", HubZigbee.deviceID);

ESP\_LOGI(TAG, "DEVICE TOKEN: %s", HubZigbee.deviceToken);

ESP\_LOGI(TAG, "STARTING CONNECT TO MQTT INNOWAY...");

ESP\_LOGI(TAG, "END PROGRAM");

}

Logging in terminal:

Text

Description automatically generated

Figure 4. Logging device information

* + 1. Compatible pair mode

This mode allows users to pair devices manually.

Function esp\_err\_t **startCompatiblePair**(deviceInfo\* device);

Input: struct deviceInfor device

Output: Các thông tin trong biến struct deviceInfor device:

Note: deviceID và deviceToken sử dụng để kết nối MQTT Innoway.

* + 1. Example test compartible mode

deviceInfo HubZigbee = {

.deviceType = "1",

.deviceName = "Hub Zigbee",

.deviceID = "",

.deviceToken = ""

};

startCompatiblePair(&HubZigbee);

ESP\_LOGI(TAG, "DEVICE ID: %s", HubZigbee.deviceID);

ESP\_LOGI(TAG, "DEVICE TOKEN: %s", HubZigbee.deviceToken);

* 1. Connecting to MQTT Innoway
     1. Description

The mqtt\_innoway library includes functions used to create connections, subscribe topics, publish and subscribe messages.

esp\_mqtt\_client\_handle\_t **mqtt\_innoway\_start**(**char** \*client\_id, **char** \*passowrd, mqtt\_innoway\_callback user\_callback);

Input:

- client\_id: is the deviceID field in struct deviceInfo, obtained after calling the startQuickPair or startCompatiblePair function.

- password: is the deviceToken field in struct deviceInfo, obtained after calling the startQuickPair or startComaptiblePair function.

- User\_callback: is the function pointer that handles MQTT client events, which the user sets.

* + 1. Example test

Source code in main.c file:

**#include** <stdio.h>

**#include** <string.h>

**#include** "esp\_wifi.h"

**#include** "nvs\_flash.h"

**#include** "esp\_event.h"

**#include** "esp\_system.h"

**#include** "esp\_log.h"

**#include** "pair\_innoway.h"

**#include** "mqtt\_innoway.h"

**static** **const** **char**\* TAG = "MAIN";

**void** **app\_main**(**void**)

{

ESP\_ERROR\_CHECK(**nvs\_flash\_init**());

// Initialize the underlying TCP/IP stack

ESP\_ERROR\_CHECK(**esp\_netif\_init**());

// Create default event loop

ESP\_ERROR\_CHECK(**esp\_event\_loop\_create\_default**());

//Creates default WIFI AP. In case of any init error this API aborts

**esp\_netif\_create\_default\_wifi\_ap**();

// Creates default WIFI STA. In case of any init error this API aborts

**esp\_netif\_create\_default\_wifi\_sta**();

// Declare variable configuration default wifi

wifi\_init\_config\_t cfg = WIFI\_INIT\_CONFIG\_DEFAULT();

// Initialize WiFi

ESP\_ERROR\_CHECK( **esp\_wifi\_init**(&cfg) );

ESP\_LOGI(TAG, STARTING TEST MQTT INNOWAY LIB");

deviceInfo HubZigbee = {

.deviceType = "1",

.deviceName = "Hub Zigbee",

.deviceID = "",

.deviceToken = ""

};

startCompatiblePair(&HubZigbee);

mqtt\_innoway\_start(HubZigbee.deviceID, HubZigbee.deviceToken, NULL);

}

* 1. Connecting to NB-IoT
     1. Description

The nb\_innoway library is compatible with the NB-IoT BC660K-GL-TE-B\_V1.1 module, it includes communication configuration APIs, network configuration, information checking, operation mode settings, APIs that support the MQTT Innoway protocol.

* + 1. Connect MCU with BC660K Module

A picture containing text, electronics, circuit

Description automatically generated

*Figure 5. Pinout BC660K Module*

MCU pin connection and BC660K module:

|  |  |
| --- | --- |
| **MCU** | **MODULE** |
| VCC | VCC |
| GND | GND |
| RXD | TXD |
| TXD | RXD |

* + 1. Example test

Run the sample code in the SDK:

<https://github.com/innoway-vht/SDK_Innoway_ESP32>

Output in Terminal should be as shown below:

Text

Description automatically generated

*Figure 6. Logging in Terminal*

References