Getting started with I-CUBE-BAIDU® IoT cloud software expansion for STM32Cube

Introduction

This user manual describes the content of the STM32Cube software expansion package for the BAIDU® IoT (Internet of things) cloud platform.
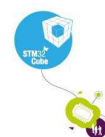
The BAIDU® IoT cloud software expansion package I-CUBE-BAIDU for STM32Cube provides application examples that connect STMicroelectronics boards to the BAIDU® IoT platform. It uses the BAIDU® iot-edge-sdk library which is compiled and running on STM32 devices.

I-CUBE-BAIDU runs on the B-L475E-IOT01 and 32L496GDISCOVERY boards.

Implementation examples are included for device-to-cloud telemetry reporting, and cloud-to- device messages for notifications to the connected devices.

I-CUBE-BAIDU offers the following features:

• Ready to run firmware example using Wi-Fi® and Cellular connectivity to support quick evaluation and development of BAIDU® IoT node applications
• Board configuration interface
• Wi-Fi® or cellular connection
• Connection to the BAIDU® IoT cloud platform
• The B-L475E-IOT01 board measures and reports the following values:
– Humidity
– Temperature
– 3D magnetic data
– 3D acceleration
– 3D gyroscope data
– Atmospheric pressure
– Proximity

Contents

List of tables

List of figures

DocIDxxxxx Rev 1

1 General information

The I-CUBE-BAIDU package for the BAIDU ® IoT cloud platform runs on STM32 32-bit microcontrollers based on the Arm® Cortex®-M processor.

Table 1 presents the definition of acronyms that are relevant for a better understanding of this document.

Table 1. List of acronyms

Term	Definition
API	Application programming interface
BSP	Board support package
CA	Certification authority
DHCP	Dynamic host configuration protocol
DNS	Domain name server
HAL	Hardware abstraction layer
IDE	Integrated development environment
IoT	Internet of things
IP	Internet protocol
LED	Light-emitting diode
RTC	Real-time clock
UART	Universal asynchronous receiver/transmitter

2   BAIDU® IoT cloud platform

This chapter introduces the BAIDU ® IoT cloud platform.
The I-CUBE-BAIDU package implements the BAIDU® iot-edge-sdk library which allows the board to securely connect to the BAIDU ® IoT cloud platform.

A user can connect to the cloud with a smartphone or personal computer and have access to the information provided by the board at any time and from any location.

Figure 1 presents the BAIDU ® IoT cloud ecosystem targeted by the I-CUBE-BAIDU package.

Figure 1. BAIDU IoT ecosystem
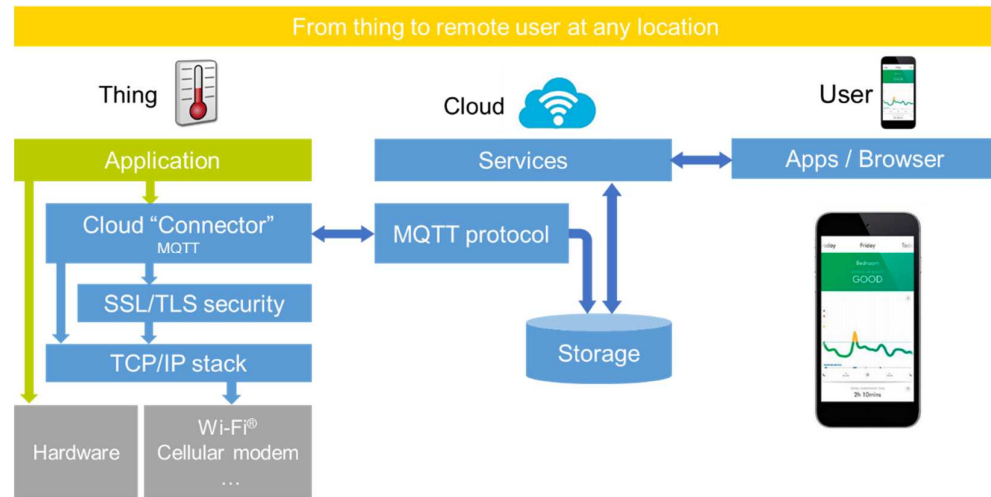
3 Package description

This chapter details the I-CUBE-BAIDU package content and the way to use it.

3.1 General description

The I-CUBE-BAIDU package provides a BAIDU® stack middleware for STM32 microcontrollers.

It is built to run on the B-L475E-IOT01 and 32F496GDISCOVERY boards and connects to the Internet through the on-board network interface:

• B-L475E-IOT01 supports Wi-Fi® connectivity with an on-board Inventek® WiFi module. This board is equipped with a set of sensors able to report humidity, temperature, 3D-axis magnetic data, 3D accelerations, 3D gyroscope data, atmospheric pressure, proximity and gesture detection (I-CUBE-BAIDU does not use the gesture detection capability).

Figure 2. B-L475E-IOT01A board

• 32L496GDISCOVERY provides a STmod+ interface that is connected to an external Quectel® UG96 cellular modem.

Figure 3 - 32L496GDISCOVERY board

The package is split into the following software components:

- Client libraries and samples for connecting to BAIDU $^®$ IoT cloud platform
- mbedTLS
- Wi-Fi$^®$ and cellular drivers
- Sensor drivers for the B-L475E-IOT01 board
- STM32L4 Series HAL
- BAIDU$^®$ application examples

The software is provided as a zip archive containing source code. The following integrated development environments are supported:

- IAR Embedded Workbench$^®$ for Arm$^®$ (EWARM)
- Keil$^®$ Microcontroller Development Kit (MDK-ARM)
- System Workbench for STM32

Note: refer to the release note available in the package root folder for information about the IDE versions supported.

3.2 Architecture

This section describes the software components of the I-CUBE-BAIDU package.

The I-CUBE-BAIDU software is an expansion for the STM32Cube. Its main features and characteristics are:
• Fully compliant with STM32Cube architecture
• Expands STM32Cube in order to enable the development of applications accessing and using the BAIDU $^®$ IoT cloud platform
• Based on the STM32CubeHAL, which is the hardware abstraction layer for STM32 microcontrollers

The software components used by the application software to access and use the BAIDU $^®$ IoT cloud platform are the following:

1. STM32Cube HAL

The HAL driver layer provides a generic multi-instance simple set of APIs (application programming interfaces) to interact with the upper layers (application, libraries and stacks).
It is composed of generic and extension APIs. It is directly built around a generic architecture and allows the layers that are built upon, such as the middleware layer, to implement their functionalities without dependencies on the specific hardware configuration for a given microcontroller unit (MCU).
This structure improves the library code reusability and guarantees an easy portability onto other devices.

2. Board support package (BSP)

The software package needs to support the peripherals on the STM32 boards apart from the MCU. This software is included in the board support package (BSP). This is a limited set of APIs which provides a programming interface for certain board specific peripherals such as the LED and the user button.

3. BAIDU$^®$ iot-edge-sdk

4. mbedTLS
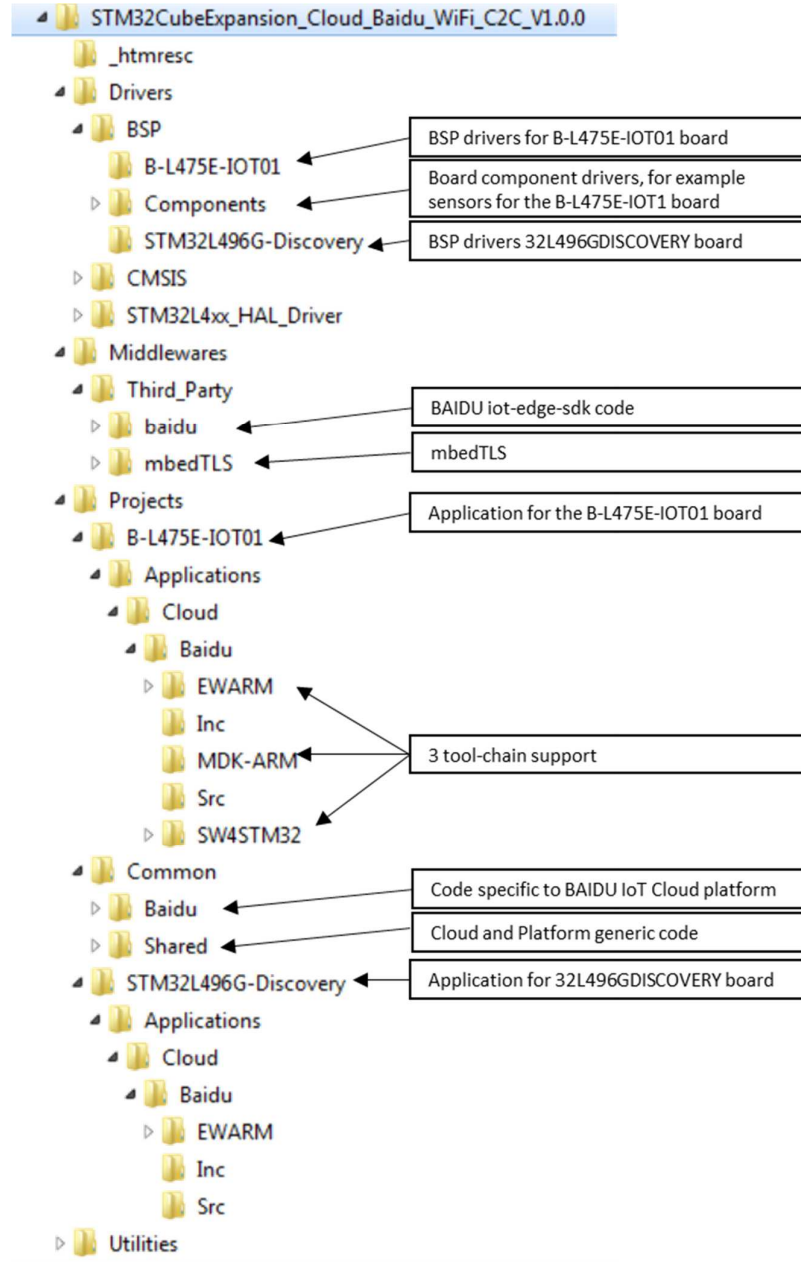
Figure 2 outlines I-CUBE-BAIDU software architecture.

Figure 4. I-CUBE-BAIDU software architecture
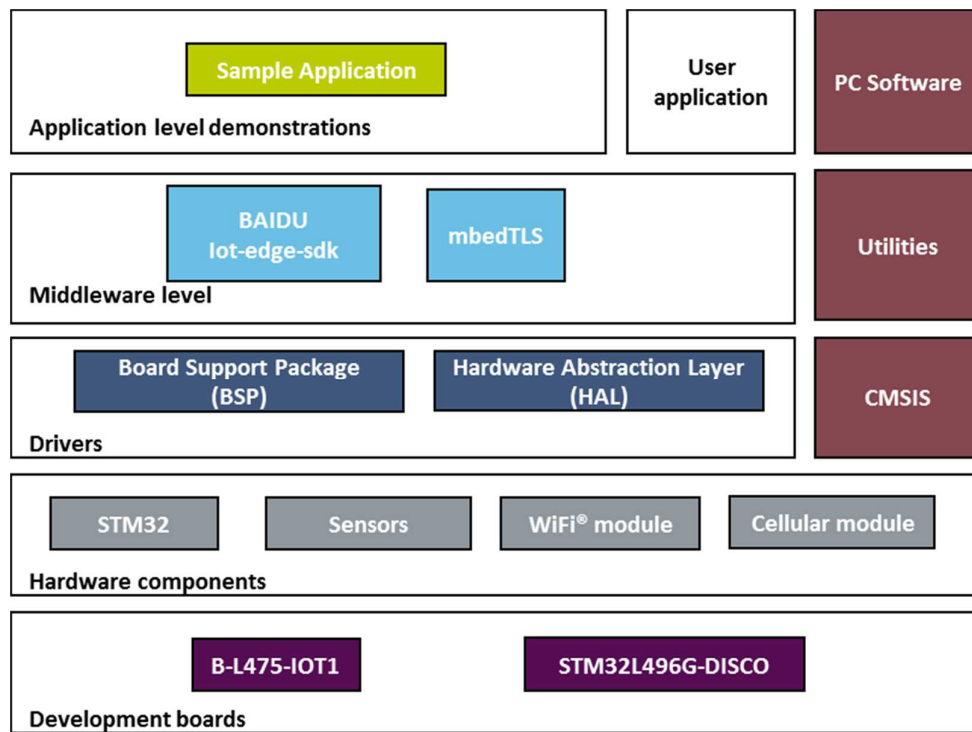
3.3 Folder structure

Figure 6 presents the folder structure of the I-CUBE-BAIDU package.

Figure 5. Project File structure

3.4 B-L475E-IOT01 board sensors

The sensors that are present on the board and used by the sample application are:
- Capacitive digital sensor for relative humidity and temperature (HTS221)
- High-performance 3-axis magnetometer (LIS3MDL)

- 3D accelerometer and 3D gyroscope (LSM6DSL)
- 260-1260 hPa absolute digital output barometer (LPS22HB)
- Proximity sensor (VL53L0X)

Example of a published sensor message:

```
{"d":
{
  "temperature": 31.39856,
  "humidity": 29.069721,
  "pressure": 997.830017,
  "proximity": 8190,
  "accX": -13,
  "accY": -14,
  "accZ": 1024,
  "gyrX": 1750,
  "gyrY": -4970,
  "gyrZ": 1470,
  "magX": 170,
  "magY": -180,
  "magZ": 605,
}
}
```

Table 2 presents the units for the values reported by the sensors of the B-L475E-IOT01 board.

Table 2. Units for the values reported by the sensors of the B-L475E-IOT01 board

Data	Unit
Temperature	degree Celsius (°C)
Humidity	relative humidity (%)
Pressure	hectopascal (hPa)
Proximity	millimeter (mm)
Acceleration	milli g-force (mgforce)
Angular velocity	millidegree per second (mdps)
Magnetic induction	milligauss (mG)

3.5 Wi-Fi® components

The Wi-Fi® software is split over Drivers/BSP/Components for the module specific software and over Projects/<board>/WiFi for I/O operations and for the Wi-Fi® module abstraction.

3.6 Reset push-button (black)

The reset push-button (black) is used to reset the board at any time. This action makes the board reboot.

3.7 User push-button (blue)

The user push-button (blue) is used in the following cases:

• To configure the Wi-Fi® and BAIDU® security credentials. This can be done from the time the board starts up and up to five seconds after that.

The application configures and manages the user button via the board support package
(BSP) functions.

The BSP functions are in the Drivers\BSP\<board name> directory.

When using the BSP button functions with the BUTTON_USER value, the application does not take into account the way this button is connected from a hardware standpoint for a given platform. The mapping is handled by the BSP.

3.8   User LED

The configuration of the user LED that is used by the applications is done via the board support package (BSP) functions.

The BSP functions are under the Drivers\BSP\<board name> directory.

Using the BSP button functions with the LED_GREEN value, the application does not take into account the way the LED is mapped for a given platform. The mapping is handled by the BSP.

3.9   mbedTLS configuration

The mbedTLS middleware is fully configurable by means of a `#include` configuration file.

The name of the configuration file can be overridden by using the following statement:

MBEDTLS_CONFIG_FILE #define.

The I-CUBE-BAIDU package uses az_mbedtls_config.h file for project configuration.

This is implemented by having the following `#` directives at the beginning of the mbedTLS.c and mbedTLS.h files:
#if !defined(MBEDTLS_CONFIG_FILE)
#include "mbedtls/config.h"
#else
#include MBEDTLS_CONFIG_FILE
#endif

The configuration file specifies the ciphers to integrate.

4 Hardware and software environment setup

To set up the hardware and software environment, one of the two supported boards must be plugged into a personal computer via a USB cable. This connection with the PC allows the user to:

• Flash the board

• Store the Wi-Fi® or the cellular credentials

• Interact with the board via a UART console

• Debug

The B-L475E-IOT01 board must be connected to a Wi-Fi® access point while the 32L496GDISCOVERY board + Quectel UC96 cellular modem must be connected to cellular network as illustrated in *Figure 4*.

Figure 6. Hardware and software setup environment
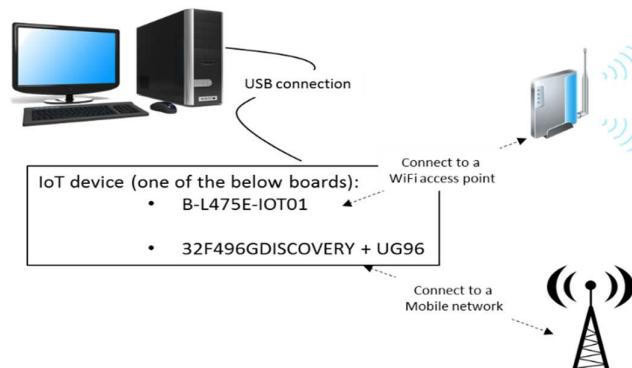
The prerequisites for running the WiFi example are:

• A MCU board: B-L475E-IOT01 (MB1297 rev D), with FW "Inventek eS-WiFi ISM43362-M3G-L44-SPI C3.5.2.3.BETA9" (Note: the FW is already included in the package and the version is displayed on the board console at boot time).

• A Wi-Fi® access point, with a transparent Internet connectivity meaning that neither a proxy, nor a firewall are blocking the outgoing traffic. It has to run a DHCP server delivering the IP and DNS configuration to the board.

The prerequisite for running the Cellular example are:

• A MCU board: STM32L496G-Discovery board equipped with a Quectel UG96 daughterboard connected on its IO expander: Modem package upside, SIM card slot downside.

• A cellular access point.

• An active cellular SIM card must be fit in the slot.

To purchase BAIDU® SIM card, you can follow the steps described in this link
https://cloud.baidu.com/doc/Pricing/IOT/14.5C.E5.8D.87.E7.BA.A7-1.html

Figure 7. Purchase BAIDU® SIM card
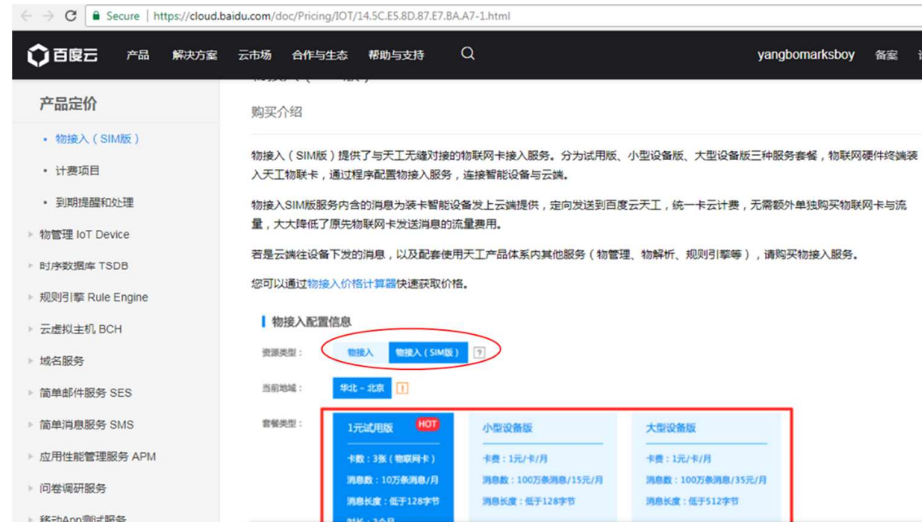
In both cases you will also need:

• A computer for running the device management application, with transparent Internet connectivity meaning that neither a proxy, nor a firewall blocking the outgoing traffic. This can for instance be the development PC, a virtual private server or a single- board computer. It can be connected to the same router as the MCU board.

• A development PC for building the application, programming through ST-Link, and running the virtual console

• A Baidu IoT cloud account to create an IoT Hub.
https://login.bce.baidu.com/?lang=en.

Figure 8. BAIDU® IoT Cloud account creation
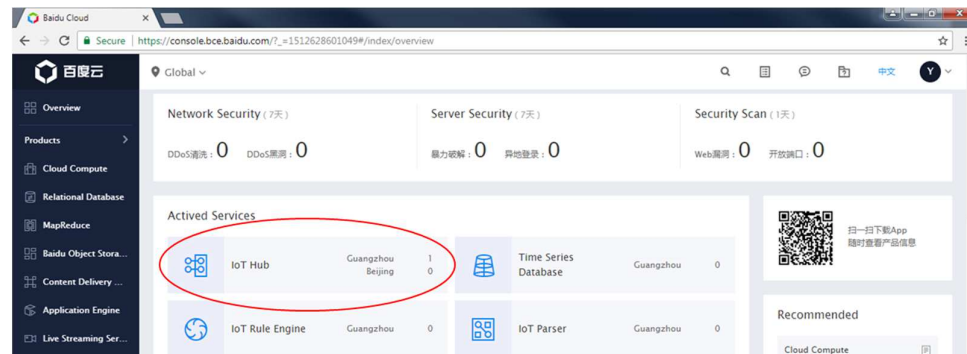
5 Interacting with the boards

A serial terminal is required to:
- Configure the WiFi AP or cellular APN credential on the node
- Display locally the received BAIDU® IoT cloud-to-device messages

The example in this document is illustrated with the use of Tera Term. Any other similar tool can be used instead.

When the board is used for the first time, it must be programmed with BAIDU® IoT device credentials (endpoint / username / password). (Please refer to *section 6.2.1* to get the BAIDU® credentials).
- Determine the STM32 ST-LINK Virtual COM port used on the PC for the Discovery board.
- Open a virtual terminal on the PC and connect it to the above virtual COM port.

A Tera Term initialization script is provided in the package utility directory (refer to *Figure 9*); this script sets the correct parameters. To use it, open Tera Term, select Setup and then Restore setup.

Note: The information provided below in this chapter can be used to configure the UART terminal as an alternative to using the Tera Term initialization script.

Terminal setup is illustrated in *Figure 9*, which shows the terminal setup and the New-line recommended parameters.

The virtual terminal New-line transmit configuration must be set to LineFeed (\n or LF) in order to allow copy-paste from UNIX type text files. The Local echo option makes copy- paste visible on the console.

Figure 9. Terminal setup
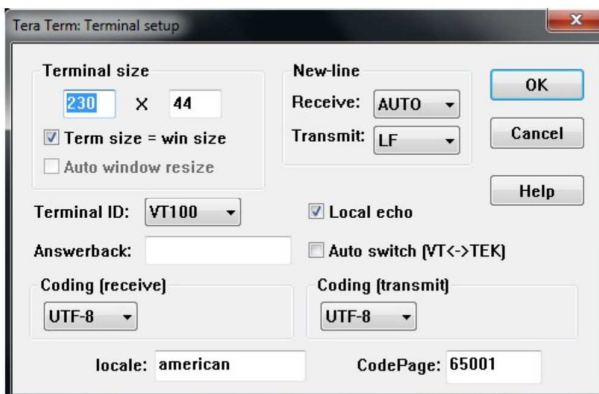
The serial port must be configured with:
- COM port number
- 115200 baud rate
- 8-bit data
- Parity none
- 1 stop bit
- No flow control

Serial port setup is illustrated in *Figure 10*.

Figure 10. Serial port setup
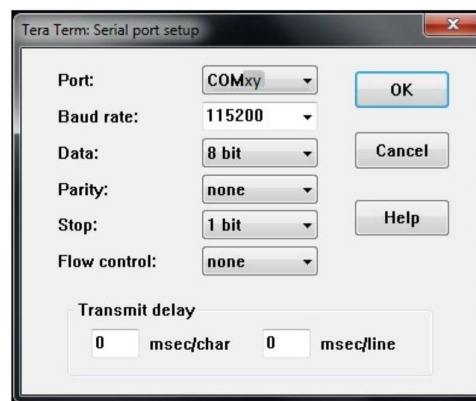
Once the UART terminal and the serial port are set up, press the board reset button (black). Follow the indications on the UART terminal to upload Wi-Fi® AP or cellular APN data. Those data remain in Flash and are reused the next time the board boots.

6 Application examples

6.1 Application description

The Baidu Sample application illustrates the various ways for a device to interact with a Baidu IoT hub, using the Baidu IoT edge SDK for C.

The application connects to a Baidu IoT hub thanks to the credentials provided by the user on the code.

6.2 Application setup

In order to make the program work, it is necessary to follow these steps:

6.2.1 Baidu device creation

Create a Baidu IoT hub instance and a device under your instance.

Figure 11. IoT hub instance

Create a device under your instance.

Figure 12. Device instance

Create a node principal to get password

Figure 13. Node principal creation
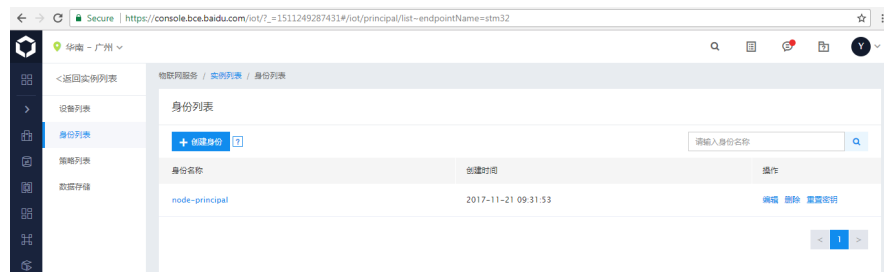
Create a node policy and topic

Figure 14. Node policy and topic setting

Set the Topic Permission

Figure 15. Topic permission setting

6.3 Application build and flash

Open the project with one of the supported development tool chains, enter the Baidu IoT hub instance address (ENDPOINT), device username and password into the source file under iothub_mqtt_client_wifi_sample.c or iothub_mqtt_client_c2c_sample.c.

Figure 16. Credential provisioning
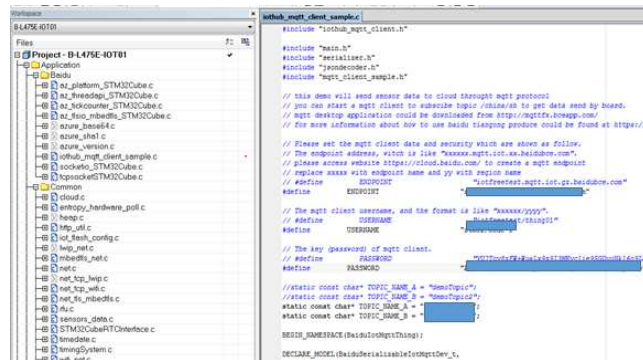
Build the project with one of the supported development toolchain

Program the firmware on the STM32 board: you can copy (or drag and drop) the generated bin file to the USB mass storage location created when you plug the STM32 board to your PC.
Alternatively, you can program the STM32 board directly through one of the supported development tool chains

6.3.1 WiFi Application first launch

The board must be connected to a PC through USB (ST-LINK USB port).
On the console of WiFi-enabled board (B-L475E-IOT01): Enter your WiFi SSID, encryption mode and password

Figure 17. WiFi credential

After the parameters are configured, it is possible to change them by restarting the board and pushing the User button (blue button) just after boot.

6.3.2 WiFi Application runtime

Once the connection to BAIDU IoT Hub is established, the board will keep publishing the on-board sensor data to the IoT Hub.

Figure 18. Data publishing

It is also possible to subscribe to the data published on the IoT Hub.

From the Web Console, you can observe the data that have been published by the node

Figure 19. WEB Console
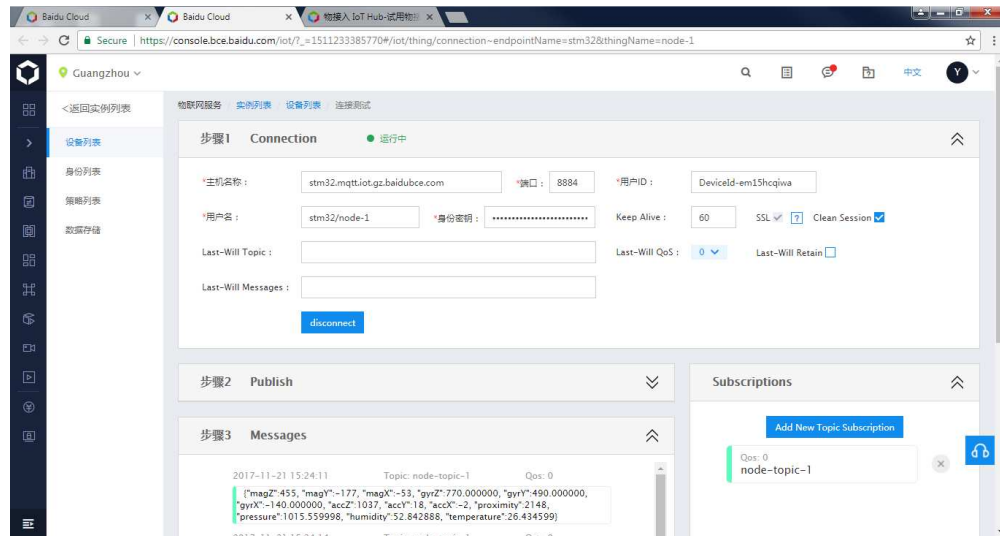
6.3.3 Cellular Application first launch

The board must be connected to a PC through USB (ST-LINK USB port)

On the console from cellular enable board (STM32L496G-Discovery board equipped with a Quectel UG96 modem), you will need to enter the C2C network configuration (SIM operator access point code, username and password).

Example:
with Emnify SIM: access point: "EM", username: "", password: ""

Figure 20. EMnify APN

with Baidu SIM: access point: "lnsyszhx01.schj.njm2mapn", username: "", password: ""
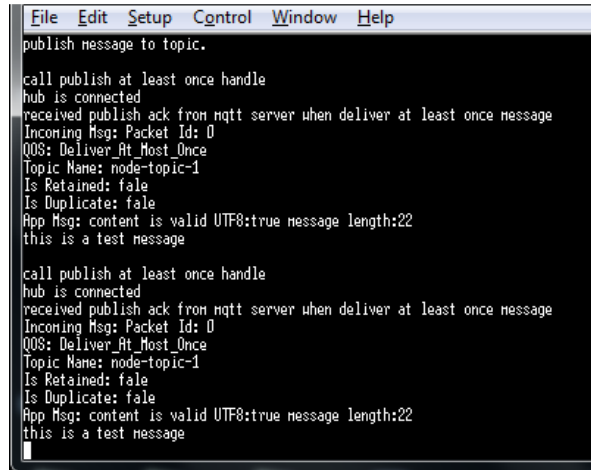
Figure 21. BAIDU APN

> IMPORTANT: Baidu SIM Card is node-locked, i.e. the Baidu APN will lock the SIM card to the node device (here ST STM32L496G-DISCO board) after the first usage. Switching the Baidu SIM card to another node devices without permission from Baidu will fail to retrieve APN and IP address during initialization stage

After the parameters are configured, it is possible to change them by restarting the board and pushing the User button (blue button) just after boot.

6.3.4 Cellular Application runtime

Once the connection to BAIDU IoT Hub is established, the board will keep publishing a "test message" text to the IoT Hub.

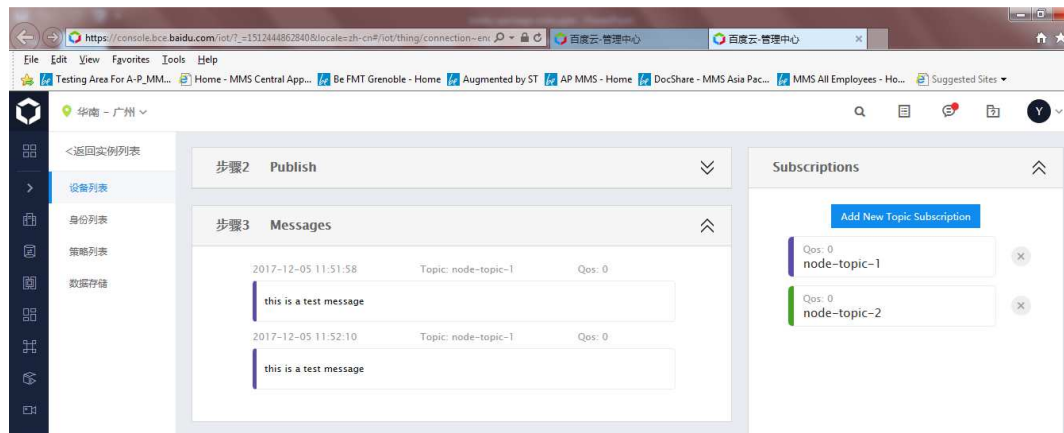**Figure 22. Cellular application runtime**

From the Web Console, you can observe the data that have been published by the node
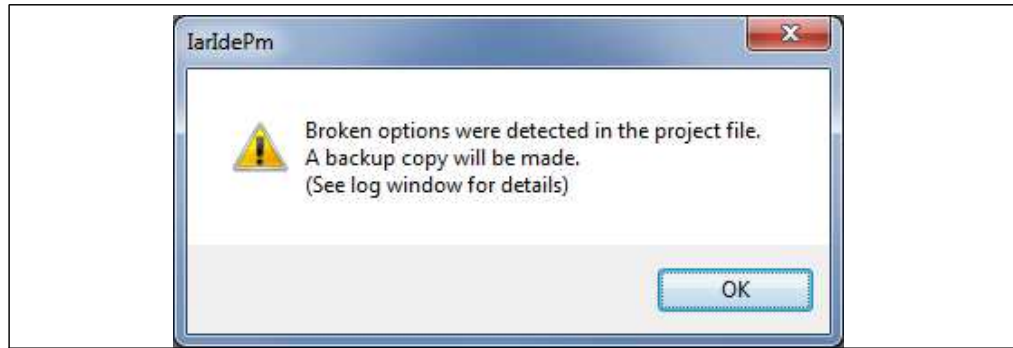
Figure 23. Web Console

7 Frequently asked questions

Q: Why do I get this pop-up (refer to *Figure 13*) when I open the project with IAR™?

Figure 24 Pop-up when the IAR™ IDE version is not compatible with the one used for I-CUBE-BAIDU

A: It is very likely that the IAR™ IDE version is older than the one used to develop the package (refer to the release note available in the package root folder for the IDE versions supported), hence the compatibility is not ensured. In this case, the IAR™ IDE version needs to be updated.

Q: How shall I modify my application to publish other messages?

A: Depending weither B-L475E-IOT01or 32L496GDISCOVERY boards are used, an update of the function PrepareMqttPayload() or PrepareToggleMqttPayload() respectively is needed in the file iothub_mqtt_client_wifi_sample.c or iothub_mqtt_client_c2c_sample.c.

Q: My device does not connect to the Wi-Fi® access point. How shall I proceed?

A: Make sure that another device can connect to the Wi-Fi® access point. If it can, enter the Wi-Fi® credentials by pressing the user button (blue) up to five seconds after board reset.

Q: The proximity sensor always reports "8190" even if I place an obstacle close to it.

A: Make sure that the liner (which is a very thin film placed on the proximity sensor) has been removed. Its color is orange and it is not very visible.

8 Revision history

Table 3. Document revision history

Date	Revision	Changes
8-Dec-2017	1	Initial release.

IMPORTANT NOTICE – PLEASE READ CAREFULLY

STMicroelectronics NV and its subsidiaries ("ST") reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information on ST products before placing orders. ST products are sold pursuant to ST's terms and conditions of sale in place at the time of order acknowledgement.

Purchasers are solely responsible for the choice, selection, and use of ST products and ST assumes no liability for application assistance or the design of Purchasers' products.

No license, express or implied, to any intellectual property right is granted by ST herein.

Resale of ST products with provisions different from the information set forth herein shall void any warranty granted by ST for such product

ST and the ST logo are trademarks of ST. All other product or service names are the property of their respective owners.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.

DocIDxxxxx Rev 1