
具有 Wi-Fi 和传感器并连接到 Amazon AWS IoT 云的 IoT 节点的 FP-CLD-AWS1 软件包入门

说明

FP-CLD-AWS1 是一款 STM32 ODE 功能包。借助该功能包，您可以直接将 IoT 节点连接到 Amazon AWS IoT 平台、发送和接收传感器数据以及发送指令至 AWS 服务或从 AWS 服务发送指令。该功能包可以快速启动任何端到端 IoT 开发，为传感器至云的开发环境所需的不同基本功能的集成节约时间。

该软件包基于 MQTT 协议实现应用级功能，并支持与 Amazon AWS IoT 平台的通信。开发人员可以使用它和 Amazon AWS 云服务来设计端到端 IoT 应用的原型，以便轻松地发送和接收实时传感器数据。

该软件在 STM32 微控制器上运行，包含用于 Wi-Fi 模块（SPWF01SA）和传感器器件（HTS221、LPS25HB、LSM6DS0、LIS3MDL、LSM6DSL 和 LSM303AGR）的驱动。

1 缩写和缩略语

表 1. 缩略语列表

缩略语	说明
AP	接入点
BSP	基础支持包
FOTA	无线固件更新
GPIO	通用输入/输出
HAL	硬件抽象层
HTML	超文本标记语言
HTTP	超文本传输协议
IDE	集成开发环境
IoT	物联网
I2C	集成电路总线
MCU	微控制器单元
MEMS	微机电系统
ODE	开放开发环境
REST API	表述性状态转移 API
SDK	软件开发套件
SMD	表面贴装器件
SSID	服务集标识符
UART	通用异步收发器
URL	统一资源定位符
WLAN	无线局域网

2 FP-CLD-AWS1 软件包概述

2.1 概述

该软件包的特性：

- 完整的中间件，基于 Wi-Fi 连接、惯性传感器和环境传感器构建应用，并将 STM32 Nucleo 板连接到 Amazon AWS IoT 云平台
- 软件接口，用于访问温度和湿度传感器（HTS221）、压力传感器（LPS25HB）和运动传感器（LIS3MDL、LSM6DS0、LSM6DSL 和 LSM303AGR）
- 集成的 Amazon AWS IoT C-SDK 软件开发套件，便于轻松访问 Amazon AWS 提供的云服务
- 连接到 NUCLEO-F401RE 板的 X-NUCLEO-IKS01A2、X-NUCLEO-IKS01A1 和 X-NUCLEO-IDW01M1 板的应用示例
- 免费友好的许可条款

该 STM32 ODE 功能包如何补充 STM32Cube？

该软件基于 STM32CubeHAL，即 STM32 微控制器的硬件抽象层。它是 STM32Cube 的扩展软件，为 Wi-Fi 和传感器扩展板提供板支持包（BSP）。驱动使底层硬件细节抽象化，这样中间件组件和应用就能以硬件独立的形式访问并控制传感器数据和 Wi-Fi 通信接口。

该软件包包含实现应用程序接口的中间件，这些接口方便基于 STM32 Nucleo 的开发套件与 Amazon 网络服务（AWS）交互。开发人员通过 Amazon AWS 云平台注册基于 STM32 Nucleo 的开发套件，即可使用该软件包来设计端到端 IoT 应用的原型，并轻松地发送和接收实时传感器数据和指令。

2.2 架构

应用软件用来访问和使用 STM32 微控制器、Wi-Fi 和传感器扩展板的软件层是：

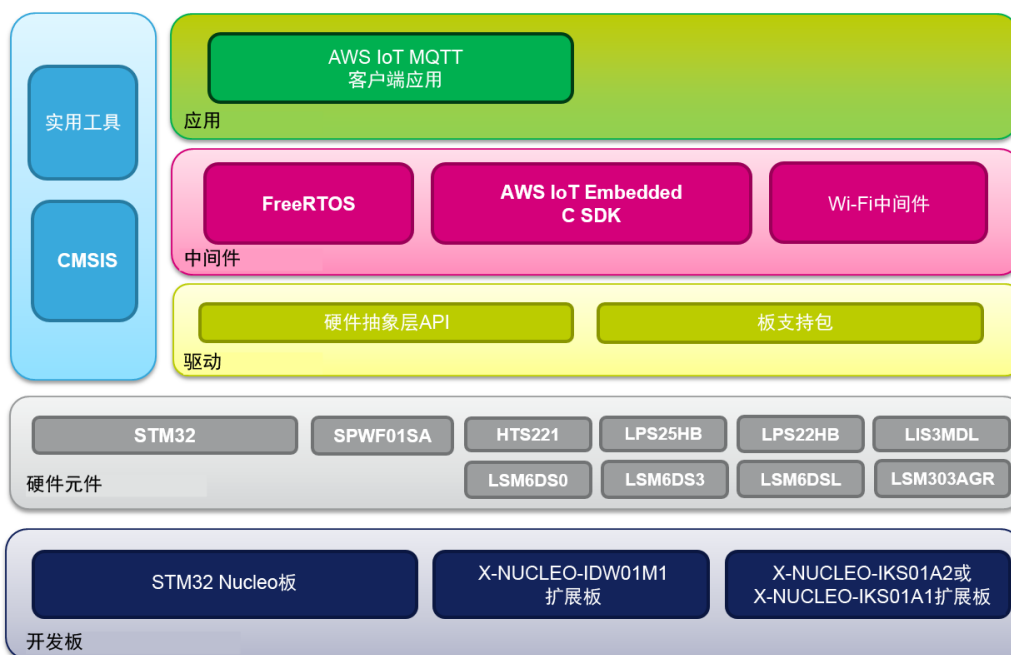
STM32Cube HAL 层：由简单、通用、多实例的一组 API（应用程序接口），与上层应用、库和堆栈交互。这些通用和扩展 API 直接围绕通用架构构建，允许在其基础上构建层（例如中间件层），实现了其功能又无需依赖给定微控制器单元（MCU）的特定硬件配置信息。此结构可提高库代码的可复用性，并确保可与其他设备轻松移植。

板级支持包（BSP）层：支持 STM32 Nucleo 板上的外设，MCU 除外。这个有限的一组 API 提供了与特定板相关外设的编程接口，如 LED、用户按钮。该接口还有助于识别板的特定版本。

中间件层：

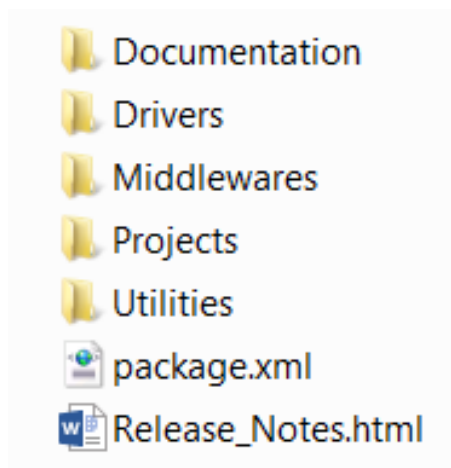
- AWS IoT Embedded C SDK（<https://github.com/aws/aws-iot-device-sdk-embedded-C>）：用于将 STM32 Nucleo 连接到 AWS IoT 平台；它集成于 STM32Cube 包中；
- RTOS 环境（FreeRTOS）：为运行 Amazon AWS IoT 设备 SDK 所需；
- SPWF01SA Wi-Fi 模块的网络接口（TCP/IP）：AWS IoT 设备 SDK 需要 TLS 协议来实现安全通信。由 SPWF01SA Wi-Fi 模块提供 TLS 接口。

图 1. FP-CLD-AWS1 软件架构



2.3 文件夹结构

图 2. FP-CLD-AWS1 软件包文件夹结构



软件包包含下列文件夹：

- **Documentation**：此文件夹包含从源码生成的已编译 HTML 文件，记录了软件组件和 API 详细信息。
- **Drivers**：包括 HAL 驱动，对于每个支持的板或硬件平台（包括那些板上组件）相关的驱动，以及 Cortex-M 系列处理器与 CMSIS 供应商无关的硬件抽象层。
- **Middlewares**：包含 Wi-Fi 的中间件接口、FreeRTOS 实时操作系统和 Amazon AWS IoT 设备 SDK。
- **Projects**：包含应用示例，可用于通过 Wi-Fi 将 STM32 Nucleo 和传感器连接到 Amazon AWS IoT。基于 NUCLEO-F401RE 平台提供了以下三个 IDE 的工程：IAR Embedded Workbench for ARM、RealView Microcontroller Development Kit（MDK-ARM）和 System Workbench for STM32。

2.4 API

用户 API 功能和参数的具体描述位于软件包 Documentation 文件夹中一个编译好的 HTML 文件中。

AWS IoT MQTT 客户端应用示例在 FreeRTOS 上名为 `void AwsIotTask (void * pvParameters)` 的单一任务中运行。

该应用示例中使用的主要 API 如下：

- `IoT_Error_t aws_iot_mqtt_init (AWS_IoT_Client *pClient, IoT_Client_Init_Params *pInitParams)`：该函数被调用来初始化 MQTT 客户端，它需要“对 IoT 客户端的引用”和“指向 MQTT 初始化的指针”作为参数。
- `IoT_Error_t aws_iot_mqtt_connect (AWS_IoT_Client *pClient, IoT_Client_Connect_Params *pConnectParams)`：该函数被调用来建立 MQTT 连接，它需要“对 IoT 客户端的引用”和“指向 MQTT 连接的指针”作为参数。
- `IoT_Error_t aws_iot_mqtt_subscribe (AWS_IoT_Client *pClient, const char *pTopicName, uint16_t topicNameLen, QoS qos, pApplicationHandler_t pApplicationHandler, void *pApplicationHandlerData)`：该函数被调用来发送 MQTT 主题的订阅请求，为了将主题和“数据”中的订阅作为实参传输给应用处理程序回调，它需要以下参数：“对 IoT 客户端的引用”、要发布的“主题名称”、“主题名称长度”和“对处理函数的引用”以及“作为参数传递给处理函数的数据”。
- `IoT_Error_t aws_iot_mqtt_yield (AWS_IoT_Client *pClient, uint32_t timeout_ms)`：该函数被调用来让当前线程让位于优先的 MQTT 客户端。MQTT 客户端利用该时间来管理 PING 请求，以便监控 TCP 连接的稳定性并定期检查套接字接收缓冲区是否存在订阅消息。必须以快于该间隔和传入消息频率的频率调用 `Yield ()`：唯有这样，客户端才能获得管理传入消息的处理时间。该函数需要参数“对 IoT 客户端的引用”和“最大毫秒数”来将线程执行传递给客户端。
- `IoT_Error_t aws_iot_mqtt_publish (AWS_IoT_Client *pClient, const char *pTopicName, uint16_t topicNameLen, IoT_Publish_Message_Params *pParams)`：该函数被调用来在主题中发布 MQTT 消息，它需要以下参数：“对 IoT 客户端的引用”、要发布的“主题名称”、“主题名称长度”和“指向发布消息的指针”。

3 Amazon AWS IoT 服务设置

3.1 概述

在运行 FP-CLD-AWS1 软件包中包含的测试应用之前，必须在 Amazon AWS IoT 中建立合适的开发环境，操作如下：

1. 创建免费的 Amazon AWS (<https://aws.amazon.com>) 帐号
2. 登录，选择 AWS IoT 服务并注册您的设备

3.2 AWS 管理控制台配置步骤

在 Amazon AWS 中创建帐户并使用凭证注册后，选择 AWS IoT 服务以进入 AWS IoT 控制台：访问和管理您的 AWS IoT 资源。

在控制台管理中：

- Step 1.** 在设备注册 (Thing Registry) 中创建设备 (<http://docs.aws.amazon.com/iot/latest/developerguide/create-device.html>)。
- Step 2.** 创建并激活设备证书 (<http://docs.aws.amazon.com/iot/latest/developerguide/create-device-certificate.html>)。
您必须离线创建您的私钥文件和证书签名请求 (CSR) 文件，并将 CSR 上传到 AWS IoT 控制台，而不是执行 AWS 文档中所述的一键式证书创建流程 (参见第 3.2.2 节 使用 OpenSSL 创建 CSR 并上传到 AWS IoT 控制台)。
- Step 3.** 创建 AWS IoT 策略 (<http://docs.aws.amazon.com/iot/latest/developerguide/create-iot-policy.html>)
如有要求，添加以下政策参数：Action : iot:* 和 Resource : *
- Step 4.** 将 AWS IoT 策略关联到设备证书 (<http://docs.aws.amazon.com/iot/latest/developerguide/attach-policy-to-certificate.html>)
- Step 5.** 将设备与证书关联 (<http://docs.aws.amazon.com/iot/latest/developerguide/attach-cert-thing.html>)
您的 STM32 Nucleo 设备现已能够与 AWS IoT 通信。

3.2.1 设备证书

使用 X.509 设备数字证书保护 STM32 Nucleo 开发板与 AWS IoT 之间的通信，该证书使用公钥基础设施 (PKI) 标准。

您可以通过 Amazon IoT 生成 X.509 证书：

1. 遵循网页 (<http://docs.aws.amazon.com/iot/latest/developerguide/create-device-certificate.html>) 上列出的标准程序，包括使用 AWS 控制台和为证书生成私钥和公钥；
2. 通过使用 AWS 控制台，离线为证书创建私钥和公钥以及证书签名请求 (CSR) 文件，然后上传至 AWS IoT 控制台

为了使用 FP-CLD-AWS1，请使用第二种方法 (参见第 3.2.2 节 使用 OpenSSL 创建 CSR 并上传到 AWS IoT 控制台)。

3.2.2 使用 OpenSSL 创建 CSR 并上传到 AWS IoT 控制台

OpenSSL 是一个开源项目，包含密码库和 SSL/TLS 工具包 (如需手册和安装说明，请访问 OpenSSL 网站：www.openssl.org)。

安装后，可使用 OpenSSL 指令生成私钥和 CSR 文件。

为了创建私钥，发出指令 `openssl ecparam -out my_ec_key.pem -name prime256v1 -genkey`：将在“my_ec_key.pem”文件中创建一个 EC 私钥。

为了创建 CSR，发出指令 `openssl req -new -config my_config.cnf -key my_ec_key.pem -out my_ec_key.csr`：将在“my_ec_key.csr”文件中创建 CSR。

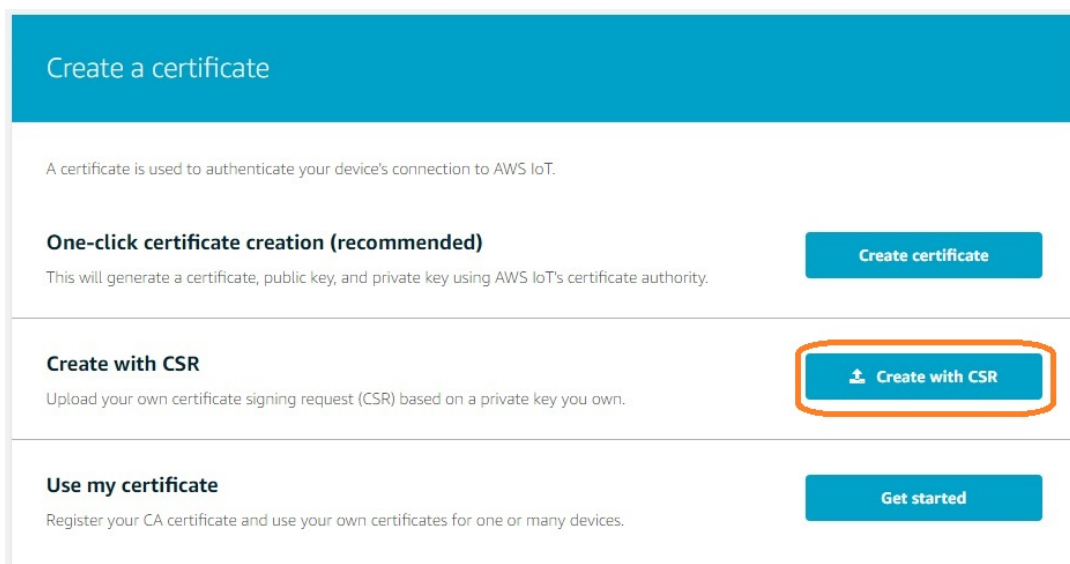
该指令要求运行一个配置文件 (cnf)，必须根据自定义信息修改该配置文件 (关于如何创建.cnf 文件的信息，请参考 OpenSSL 文档)。

下面是一个配置文件示例：

```
[req]
prompt = no
distinguished_name = dn
req_extensions = ext
[dn]
CN = www.xxx.com
emailAddress = xxx@xx.com
O = XXX
L = XXX
C = XX
[ext]
```

创建后，必须在 AWS IoT 控制台中更新 CSR 文件：点击“使用 CSR 创建”按钮，使用之前通过 OpenSSL 生成的 .csr 文件为您的设备创建证书。

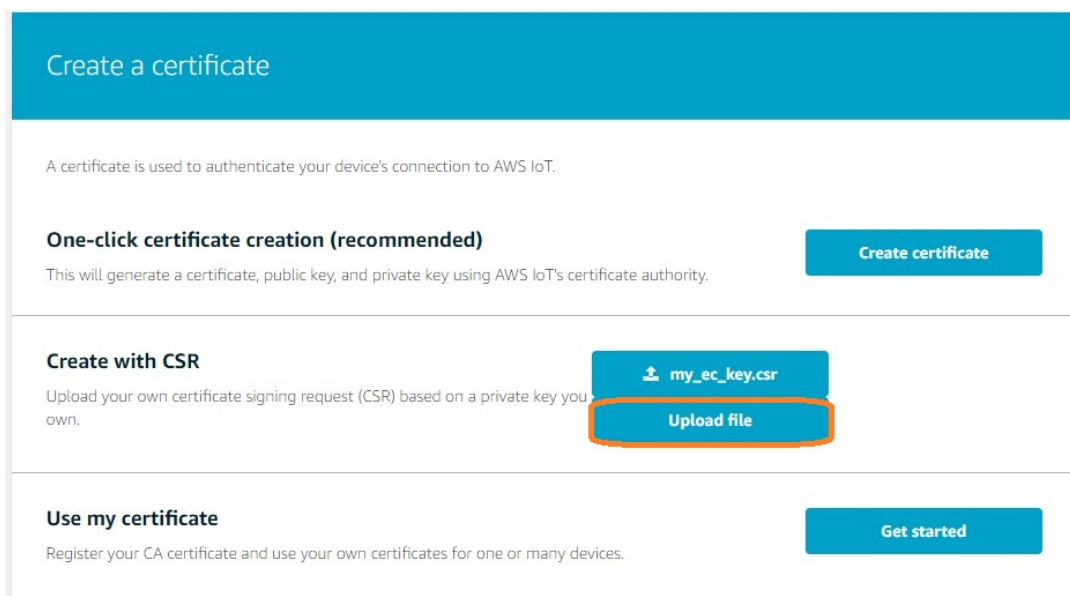
图 3. 在 AWS 控制台中通过 CSR 文件创建证书



The screenshot shows the 'Create a certificate' page in the AWS IoT console. It has a blue header with the title 'Create a certificate'. Below the header, there is a brief explanation: 'A certificate is used to authenticate your device's connection to AWS IoT.' There are three main sections, each with a button on the right:

- One-click certificate creation (recommended)**: This will generate a certificate, public key, and private key using AWS IoT's certificate authority. Button: 'Create certificate'.
- Create with CSR**: Upload your own certificate signing request (CSR) based on a private key you own. Button: 'Create with CSR' (highlighted with an orange rectangle).
- Use my certificate**: Register your CA certificate and use your own certificates for one or many devices. Button: 'Get started'.

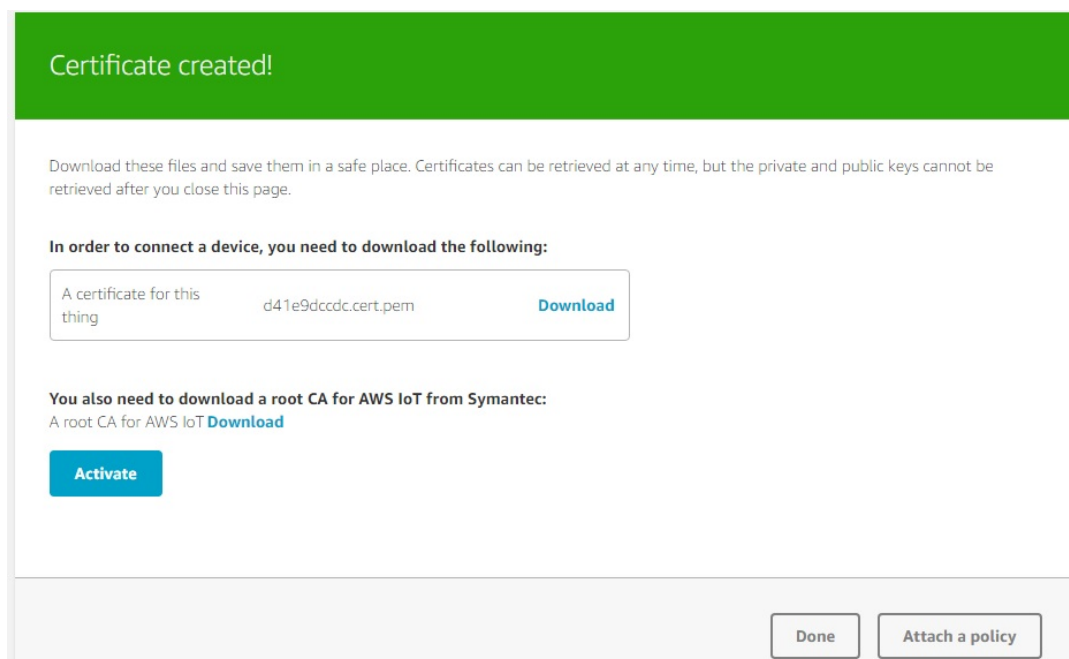
图 4. 将 CSR 文件上传到 AWS 控制台



This screenshot is similar to the previous one, showing the 'Create a certificate' page. In this view, the 'Create with CSR' section is expanded, showing a file upload interface. A file named 'my_ec_key.csr' is shown with an upload icon. Below the file name, the 'Upload file' button is highlighted with an orange rectangle. The other sections and buttons remain the same as in the previous screenshot.

随即创建证书:

图 5. 成功创建证书



4 FP-CLD-AWS1 示例应用描述

X-NUCLEO-IKS01A2 或 X-NUCLEO-IKS01A1 和 X-NUCLEO-IDW01M1 扩展板的应用示例，**Projects**”目录下提供了使用 NUCLEO-F401RE 开发板。

包括多个 IDE 的工程。

该应用从温度、湿度、压力、加速度计、磁力计和陀螺仪传感器读取数据值，并通过 Wi-Fi 将其发送至 Amazon AWS IoT。

为了运行应用，如第 3 节 **Amazon AWS IoT 服务设置** 所述设置 Amazon AWS 服务。然后，在重新编译项目前，需要更新软件包源代码中的若干细节（参见第 4.1 节 **通过 AWS IoT 配置更新源代码**）。当二进制代码就绪并刷入到 Nucleo 板时，可以按照第 4.2 节 **配置 Wi-Fi 接入点** 参数并将 STM32 Nucleo 板连接到 AWS IoT 控制台所述启动应用。

4.1 通过 AWS IoT 配置更新源代码

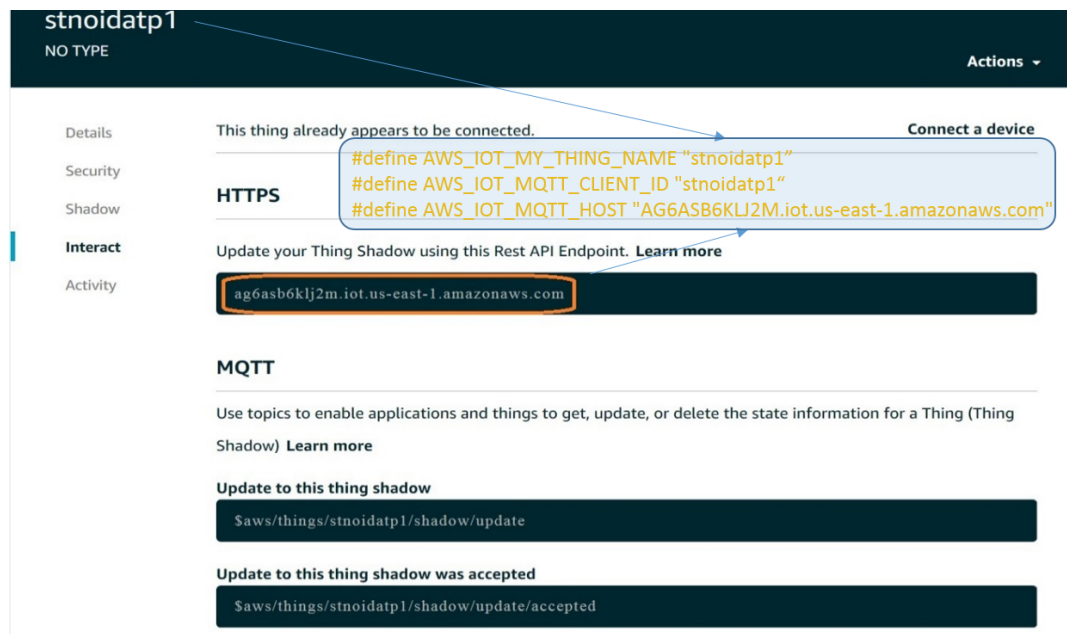
本节描述如何使用 Amazon 凭证和设备证书更新 FP-CLD-AWS1 源文件。

4.1.1 aws_iot_config.h 中的更改

必须更新 **Middlewares/Third_Parties/aws-sdk/samples/nucleo/sub_pub/aws_iot_config.h** 中的以下“defines”：

- **AWS_IOT_MY_THING_NAME**：与 Nucleo 设备关联并在 AWS IoT 控制台中创建（如第 3 节 **Amazon AWS IoT 服务设置** 所述）的设备名称。例如：`#define AWS_IOT_MY_THING_NAME "stnoidatp1"`
- **AWS_IOT_MQTT_CLIENT_ID**：每个设备的 MQTT 客户端 ID 应是唯一的；它可以与设备名称相同。例如：`#define AWS_IOT_MQTT_CLIENT_ID "stnoidatp1"`
- **AWS_IOT_MQTT_HOST**：通过选择设备的详细信息选项卡，可以从 AWS IoT 控制面板获得（参见下图）。例如：`#define AWS_IOT_MQTT_HOST "AG6ASB6KLJ2M.iot.us-east-1.amazonaws.com"`。通过选择“注册”（Registry）=>“设备”（Things）=> **stnoidatp1** =>“交互”（Interact），可以从 AWS IoT 控制面板获得。

图 6. AWS IoT 控制面板：设备详细信息



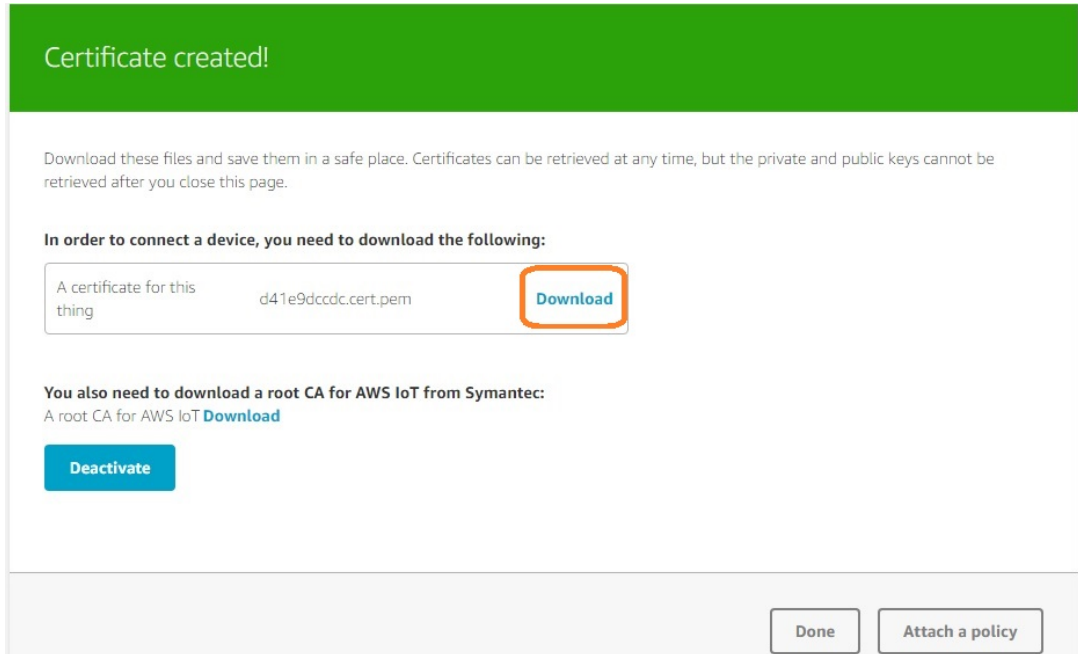
4.1.2 aws_nucleo_certificates.c 中的更改

文件 **Middlewares/Third_Parties/aws-sdk/platform/nucleo/aws_nucleo_certificates.c** 包含两个要更新的变量：`uint8_t clientCRT []`，设备身份证书；`uint8_t clientKey[]`，设备私钥。

`uint8_t clientCRT []`，设备身份证书

该变量包含 X.509 设备证书，通过选择证书图标，可以从 AWS IoT 控制台下载该证书，如下图所示：

图 7. 证书下载



下载后，从 AWS 接收的文件包含具有如下格式的数据：

```
-----BEGIN CERTIFICATE-----
MIIC2DCCAcCgAwIBAgIVAKX1mBt63N3G39cAnz7U7JX1Hd6DMA0GCSqGSIb3DQEB
CwUAME0xSzbJBgNVBAsMQkFtYXpvaW50dVYdmljZXMGTz1BbWF6b24uY29t
IEluYy4gTD1TZWF0dGx1IFNUPVdhc2hpbmd0b24gQz1VUzAeFw0xNjAzMDgwODQ3
MTNaFw00OTEyMzEyMzU5NTlaMGcxZzARBgNVBAMMNd3dy5zdC5jb20xJDAiBgkq
hkiG9w0BCQEFWFudXAta3VtYXJzZGFzQHN0LmNvbTEMMAoGA1UECgwDU1RNMQ8w
DQYDVQQHDAZHTm9pZGExCzAJBgNVBAYTAklOMFkwEwYHKoZIzj0CAQYIKoZIzj0D
AQcDQgAEbTVHX1Clq0KebzauSpNSvFFy4Dj+pcfKyNVtquy1zrVkuoQJjNHQjY0
smphQv/y7xv1XMF96BaCb6E4K8VXaNgMF4wHwYDVR0jBBGwFoAU6XIqHcA8mtSX
JyqxwPGn6hS9DWIwHQYDVR0OBBYEFao3NtsRZhOto7YJUCHKUJ62fuAHMAwGA1Ud
EwEB/wQCMAAwDgYDVDR0PAQH/BAQDAgeAMA0GCSqGSIb3DQEBBCwUAA4IBAQAf2oVC
dyo+V/+lDZh+A9hxSWI85hIGL4o6GWHbqgCtZGIPTU5kKwATf10QjSmxTXbYSSFD
18c1OJNCvG0EA8t9ScHPSzOY6hJ+mW0uxd8emln2oWjCpLmvHPUWwnnnijXPnz9t
x73jNVxJCGLUk01I22hzHetV2cFxyBhKh9nPSagxKIwJye9qL7I6yLqgfbKvr5Ob
7Z7lDoeIHyXsQ5jjTCFqjR7hJ47v8WKJ0Os87VV2stGUQW5CAHnk+oZyaXqnSiYB
USsyEx2MA7asEs237BpiNOx0RhBhVuP/CtafigBTBqubH8yHV4NclCLnigW+UzTA
BseHclKIGJJK3bnM
-----END CERTIFICATE-----
```

该 X.509 证书是一串字节，保存在一个 TXT 格式文件中。在 TXT 文件中每一行字节的末尾，都有一个换行符“\n”。

FP-CLD-AWS1 软件包不在节点中直接使用 TXT 格式的证书文件。它使用文件中的数据作为无符号字符数组（C 语言编程结构）。为了在数组中保存数据，我们需要额外的字符来满足 X.509 证书结构和 C 语言编程要求。

X.509 证书格式需做如下更改：

- 在-----BEGIN CERTIFICATE-----字符串行的后面添加\n
- 在-----END CERTIFICATE-----字符串行的前面添加\n
- 在-----END CERTIFICATE-----字符串行的后面添加\n

为了满足 C 语言编程要求，唯一需要的更改是在每一行字符串的末尾添加反斜杠字符（\）。

按以下格式复制数组中的数据：

```
uint8_t clientCRT[] = "\
-----BEGIN CERTIFICATE-----\n\
MIIC2DCCAcCgAwIBAgIIVAKX1mBt63N3G39cAnz7U7JX1Hd6DMA0GCSqGSIb3DQEB\
CwUAME0xSzBjBjNVBAsMQkFtYXpvbiBXZWIgU2VydmljZXMgTz1BbWF6b24uY29t\
IEluYy4gTD1TZWF0dGx1IFNUPVdhc2hpbmd0b24gQz1VUzAeFw0xNjAzMDgwODQ3\
MTNaFw00OTeyMzEyMzU5NTlaMGcxZzARBjNVBAMCnd3dy5zdC5jb20xJDAiBgkq\
hkiG9w0BCQEWFWFudXAta3VtYX1uZGFzQHN0LmNvbTEuMGA1UECgwDU1RNMQ8w\
DQYDVQQHDAZHTm9pZGExCzAJBgNVBAYTAklOMFkwEwYHKoZIzj0CAQYIKoZIzj0D\
AQcDQgAEbTVHX1Clq0KezbzauSpNSvFFy4Dj+pcfKyNVtquy1zrVkuUoQJjNHQjY0\
smphQv/y7xv1XMZF96BaCb6E4K8VXaNgMF4wHwYDVR0jBBgwFoAU6XIqHcA8mtSX\
JyqxwPGn6hS9DWIwHQYDVR0OBBYEFaO3NTsRZhOto7YJUCHKUJ62fuAHMAwGA1Ud\
EwEB/wQMAAwDgYDVR0PAQH/BAQDAgeAMA0GCSqGSIb3DQEBBCwUAA4IBAQAf2oVC\
dyo+V/+1DZh+A9hxSWI85hIGL4o6GWHbqgCtZGIPTU5kKwATf10QjSmxTXbYSSFD\
18clOJNCvG0EA8t9ScHPSzOY6hJ+mW0uxd8em1n2oWjCpLmvHPUWwnnnijXPNZ9t\
x73jNVxJCGLUk01I22hzHetV2cFxyBhKh9nPSagxKIwJye9qL7I6yLqgfbKvr5Ob\
7Z71DoeIHxS5Q5jjTCFqj7hJ47v8WKJ00s87VV2stGUQW5CAHnk+oZyaXqnSiYB\
USsyEx2MA7asEs237BpiNOx0RhBhVuP/CtafigBTBqubH8yHV4NclCLnigW+UzTA\
BseHclKIGJJK3bnM\
\n-----END CERTIFICATE-----\n
```

uint8_t clientKey[], 设备私钥

该变量包含使用 OpenSSL 生成的设备私钥（第 3.2.2 节 使用 OpenSSL 创建 CSR 并上传到 AWS IoT 控制台所述的 .pem 文件）。

```
-----BEGIN EC PARAMETERS-----
BggqhkjOPQMBBw==
-----END EC PARAMETERS-----
-----BEGIN EC PRIVATE KEY-----
MHcCAQEEICASgpjAQK+JTScxRh6f8fTZ5Q1SYXDqiskyRPMpx2JpoAoGCCqGSM49
AwEHoUQDQgAEbTVHX1Clq0KezbzauSpNSvFFy4Dj+pcfKyNVtquy1zrVkuUoQJjNH
QjY0smphQv/y7xv1XMZF96BaCb6E4K8VXQ==
-----END EC PRIVATE KEY-----
```

如上文 uint8_t clientCRT [], 设备身份证书中所述，该私钥也是以 TXT 格式文件生成的。因此，要完成的更改与上文列出的类似。

X.509 证书格式需做如下更改：

- 在-----BEGIN EC PARAMETERS-----字符串行的后面添加\n
- 在-----END EC PARAMETERS-----字符串行的前面添加\n
- 在-----END EC PARAMETERS-----字符串行的后面添加\n
- 在-----BEGIN EC PRIVATE KEY-----字符串行的后面添加\n
- 在-----END EC PRIVATE KEY-----字符串行的前面添加\n
- 在-----END EC PRIVATE KEY-----字符串行的后面添加\n

为了满足 C 语言编程要求，唯一需要的更改是在每一行字符串的末尾添加反斜杠字符（\）。

修改变量 uint8_t clientKey[]：

```
uint8_t clientKey[] = "\
-----BEGIN EC PARAMETERS-----\n\
BggqhkjOPQMBBw==\
\n-----END EC PARAMETERS-----\n\
-----BEGIN EC PRIVATE KEY-----\n\
MHcCAQEEICASgpjAQK+JTScxRh6f8fTZ5Q1SYXDqiskyRPMpx2JpoAoGCCqGSM49\
AwEHoUQDQgAEbTVHX1Clq0KezbzauSpNSvFFy4Dj+pcfKyNVtquy1zrVkuUoQJjNH\
QjY0smphQv/y7xv1XMZF96BaCb6E4K8VXQ==\
\n-----END EC PRIVATE KEY-----\n"
```

4.2

配置 Wi-Fi 接入点参数并将 STM32 Nucleo 板连接到 AWS IoT 控制台

在 AWS IoT 云设置完毕并完成硬件设置的配置（参见第 5 节 系统设置指南）后，使用串口终端显示来自基于 STM32 Nucleo 的系统的信息。

必须为串行终端配置以下参数：

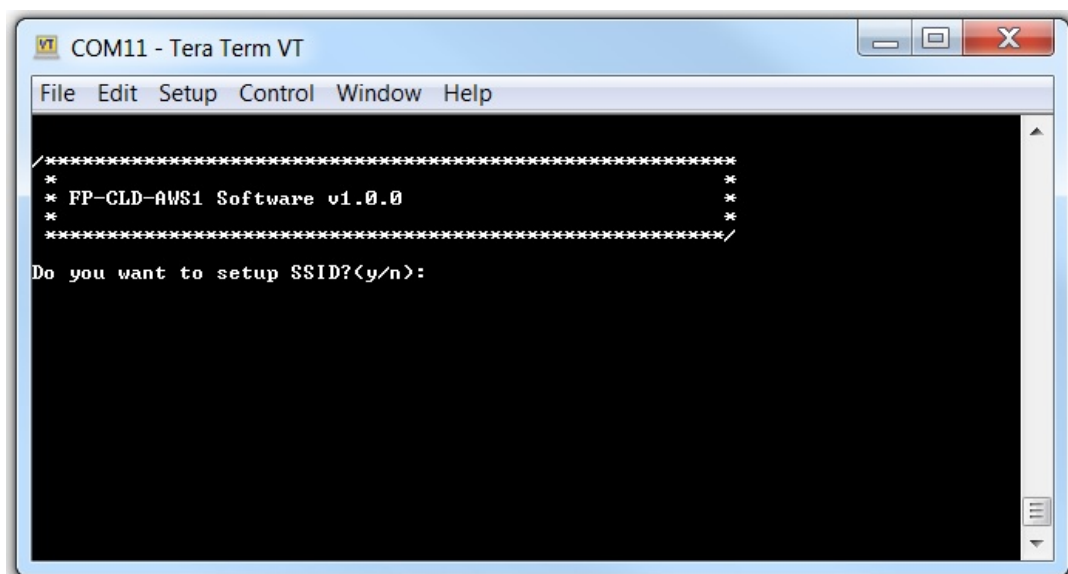
- 波特率：460800
- 数据：8 位
- 校验位：无
- 停止：1 位
- 流控制：无
- 新行接收：CR+LF
- 新行发送：CR

为了通过 Wi-Fi 接入点（AP）连接基于 STM32 Nucleo 的系统，必须通过在串行接口调试工具中输入来配置相应的 SSID 和密码。

通过串行接口调试工具配置接入点参数

- 复位 STM32 Nucleo 板；
- 等待系统提示您在串行接口调试工具中输入 SSID 和密码，如下图所示：

图 8. 通过串行接口调试工具设置 SSID 和密码



- 输入 SSID 和密码；
- 应用从串行接口读取参数并用其连接 AP。

在 STM32 Nucleo 板连接到 Wi-Fi 接入点后，MQTT 客户端应用自动联系 AWS IoT 控制台。按下 STM32 Nucleo 板上的用户按钮（蓝色按钮），来发布和订阅传感器数据。

可在串口终端中查看传感器数据记录。

图 9. 终端传感器数据查看

```

COM11 - Tera Term VT
File Edit Setup Control Window Help
-->sleep
.....Push User Button to Publish Sensors Data...
-->sleep

Publish MQTT Packet.....
Temperature: 29.00
Humidity : 45.50
Pressure : 987.47
ACC-X : 16 ACC-Y: 8 ACC-Z: 987
GYR-X : 630 GYR-Y: -910 GYR-Z: -2660
MAG-X : -28 MAG-Y: 76 MAG-Z: -142

Subscribe MQTT Packet.....
Temperature: 29.00
Humidity : 45.50
Pressure : 987.47
ACC-X : 16 ACC-Y: 8 ACC-Z: 987
GYR-X : 630 GYR-Y: -910 GYR-Z: -2660
MAG-X : -28 MAG-Y: 76 MAG-Z: -142

-->sleep
.....Push User Button to Publish Sensors Data...

```

AWS IoT 控制台接收的传感器数据可用于开发依赖于 Amazon AWS IoT 的云应用，并构建端到端 IoT 系统。

5 系统设置指南

5.1 硬件说明

本节概括了将基于 STM32 Nucleo 的平台连接至 Amazon AWS IoT 控制台所需的硬件元件。

5.1.1 STM32 Nucleo 平台

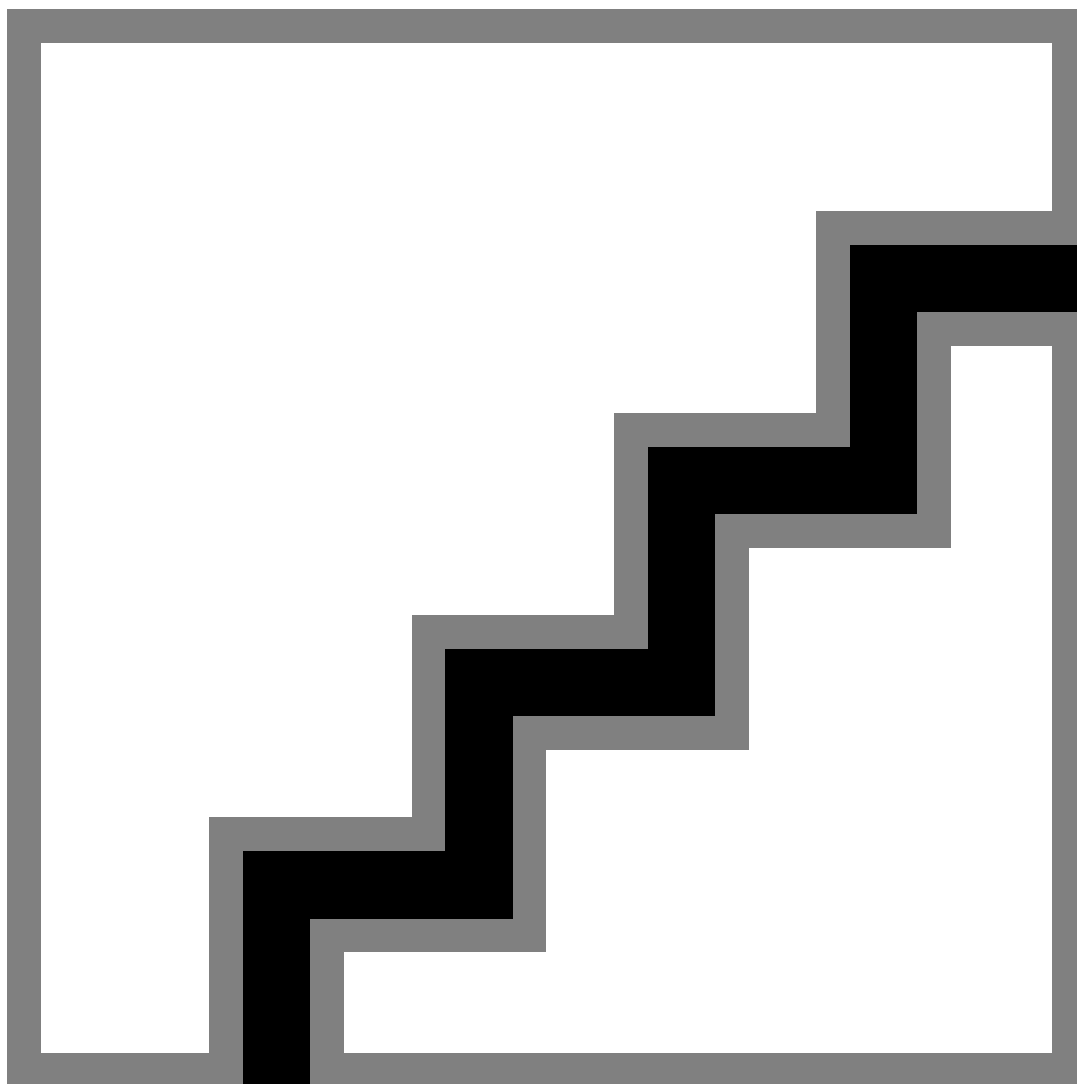
STM32 Nucleo 开发板为用户提供了经济、灵活的方法来测试解决方案，构建任何 STM32 微控制器产品线的原型。

Arduino™连接支持和 ST Morpho 连接器使之能够轻松使用各种专用扩展板扩展 STM32 Nucleo 开放式开发平台的功能。

由于集成了 ST-LINK/V2-1 调试/编程器，STM32 Nucleo 板无需另外加调试器。

STM32 Nucleo 板附带全面的 STM32 软件 HAL 库和各种套装软件示例。

图 10. STM32 Nucleo 板



关于 STM32 Nucleo 板的信息请参考 www.st.com/stm32nucleo

5.1.2 X-NUCLEO-IDW01M1 扩展板

X-NUCLEO-IDW01M1 是一种基于 **SPWF01SA** 模块的 Wi-Fi 评估板，用于扩展 **STM32 Nucleo** 板。

CE、IC 和 FCC 认证的 **SPWF01SA** 模块内嵌 **STM32 MCU**、带集成功率放大器和功率管理功能的低功耗 **Wi-Fi b/g/n SoC** 以及 **SMD 天线**。

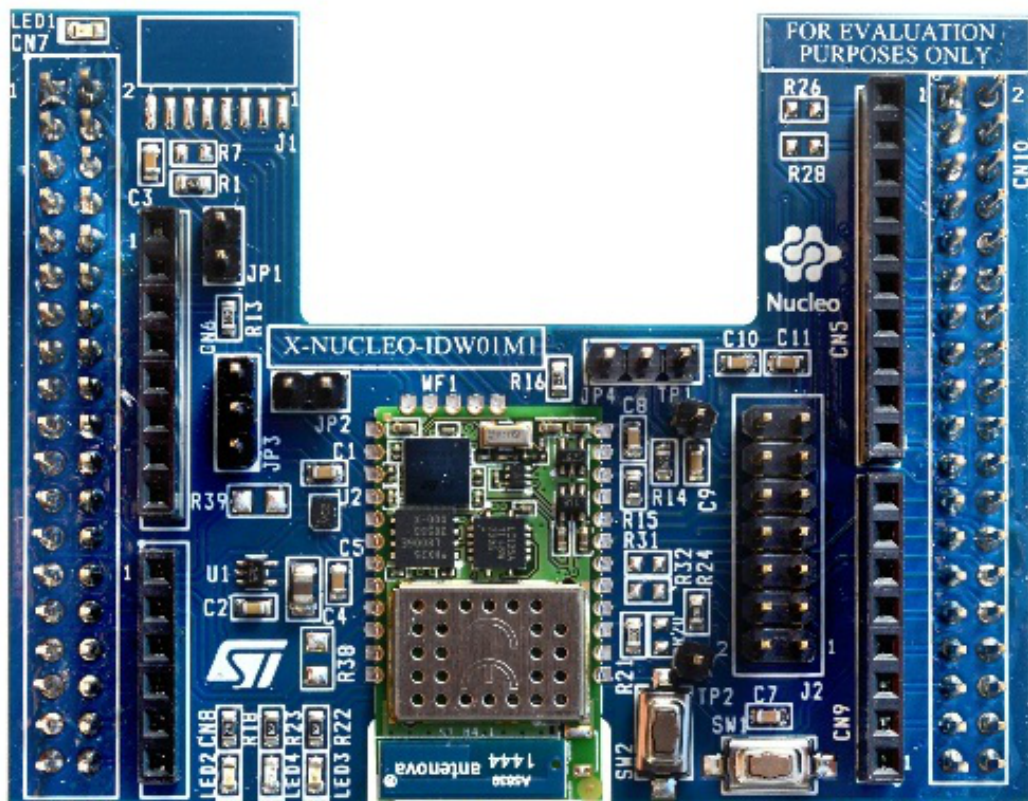
SPWF01SA 模块还配备 **1 MB 外部闪存**，用于固件的无线更新（**FOTA**）。

固件具有可打开最多 **8 个 TCP/UDP 套接字** 的完整软件 **IP 栈**，用于与模块交互的具有 **SSI** 的动态网页，以及为方便将文件传输至云中的服务器或从云中的服务器传输文件的 **REST API**（获取和提交）。模块可以同时充当套接字服务器和套接字客户端。

固件支持通过 **TLS/SSL** 加密实现套接字安全，确保与云进行安全的端到端通信（有或没有身份验证）。模块作为客户端 **STA**、**IBSS** 或 **miniAP** 工作（有最多 **5 个客户端 STA**）。

X-NUCLEO-IDW01M1 通过 **UART** 串行端口与 **STM32 Nucleo** 板上的 **MCU** 连接；用户可以轻松地使用 **AT** 指令访问栈函数。**X-NUCLEO-IDW01M1** 与 **ST morpho** 和 **Arduino UNO R3** 接口兼容。

图 11. X-NUCLEO-IDW01M1 Wi-Fi 扩展板

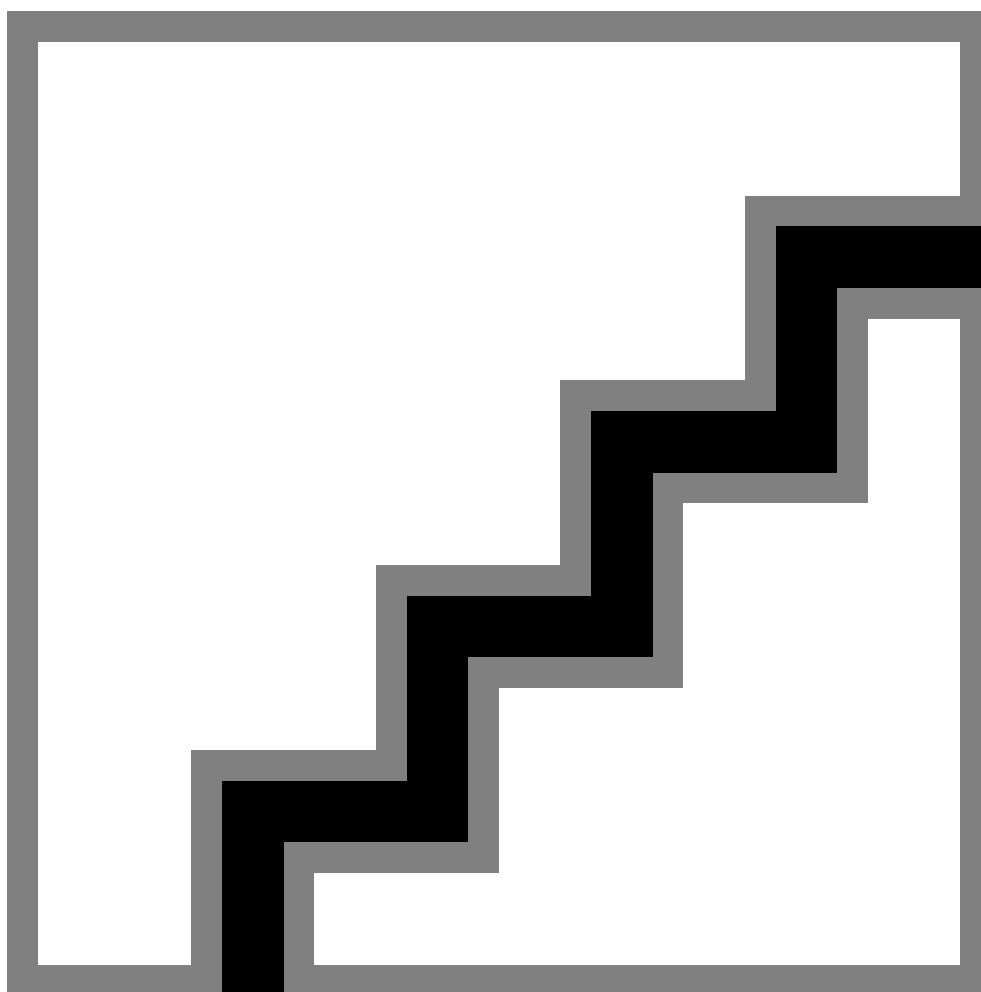


关于扩展板的信息，请参考 [www.st.com](http://www.st.com/x-nucleo)（<http://www.st.com/x-nucleo>）。

5.1.3 X-NUCLEO-IKS01A1 扩展板

X-NUCLEO-IKS01A1 是传感器扩展板，用于 **STM32 Nucleo** 板。它还兼容 **Arduino UNO R3** 接口，并围绕湿度（**HTS221**）、压力（**LPS25HB**）和运动（**LIS3MDL** 和 **LSM6DS0**）传感器进行设计。**X-NUCLEO-IKS01A1** 通过 **I²C** 引脚与 **STM32 MCU** 连接，用户可通过更改评估板上的一个电阻器来更改默认 **I²C** 端口和设备 **IRQ**。可以连接 **LSM6DS3 DIL24** 扩展元件并用它替代某个 **LSM6DS0** 传感器。

图 12. X-NUCLEO-IKS01A1 扩展板



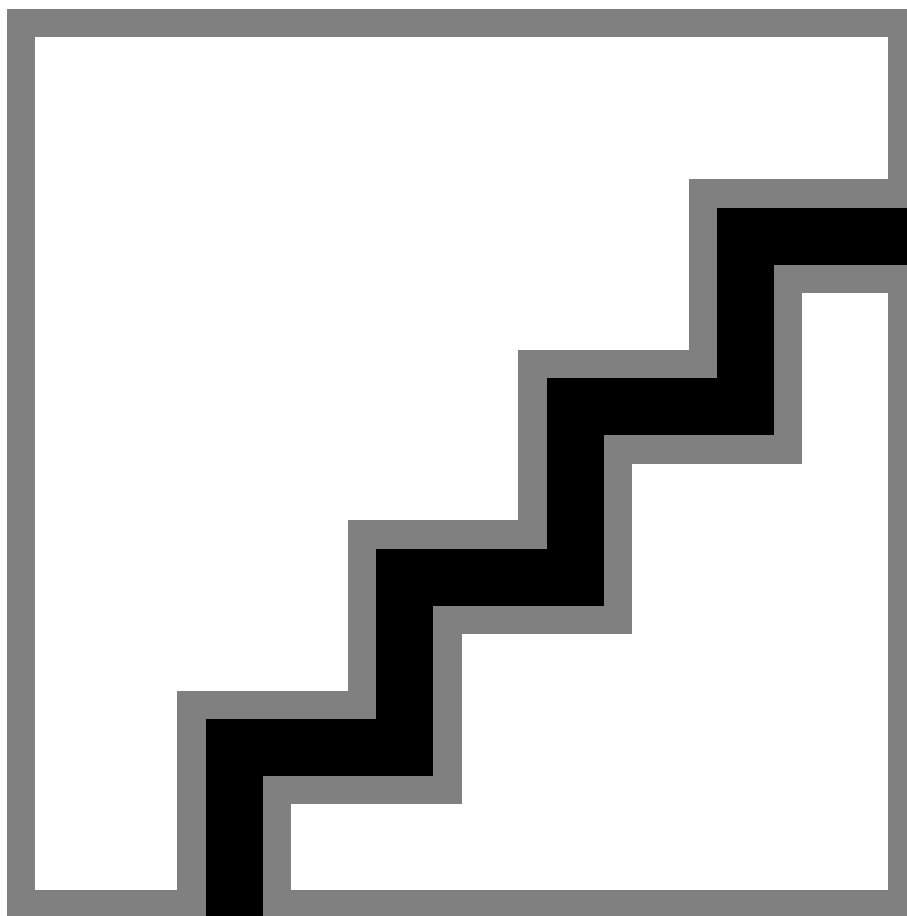
5.1.4 X-NUCLEO-IKS01A2 扩展板

X-NUCLEO-IKS01A2 为 STM32 Nucleo 的运动 MEMS 和环境传感器扩展板。

它与 Arduino UNO R3 接口兼容，并设计为与 LSM6DSL 3D 加速度计和 3D 陀螺仪、LSM303AGR 3D 加速度计和 3D 磁力计、HTS221 湿度和温度传感器以及 LPS22HB 压力传感器共同工作。

X-NUCLEO-IKS01A2 与 STM32 微控制器通过 I²C 引脚进行连接，可以更改默认的 I²C 端口。

图 13. X-NUCLEO-IKS01A2 MEMS 和环境传感器扩展板



5.2 硬件设置

需要下列硬件组件：

1. STM32 Nucleo 开发平台（订购代码：NUCLEO-F401RE）
2. Wi-Fi 扩展板（订购代码：X-NUCLEO-IDW01M1）
3. 传感器扩展板（订购代码：X-NUCLEO-IKS01A2 或 X-NUCLEO-IKS01A1）
4. Type A USB 转 Mini-B USB 连接线，用于连接 STM32 Nucleo 和 PC

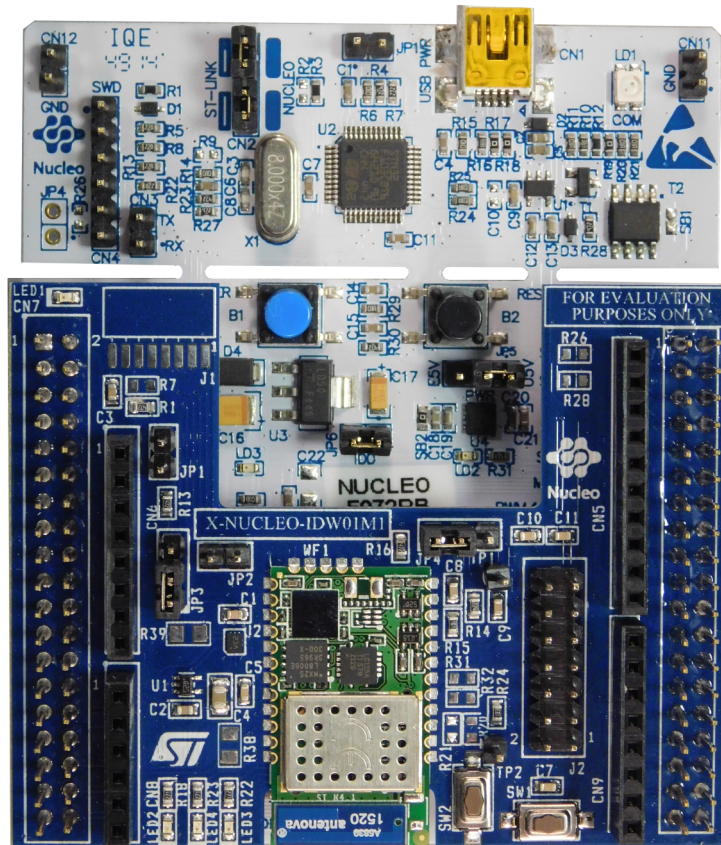
5.2.1 STM32 Nucleo 和扩展板设置

STM32 Nucleo 板集成 ST-LINK/V2-1 调试工具/编程工具

开发人员可以从网站（www.st.com）上下载相关的 ST-LINK/V2-1 USB 驱动版本（对于 Windows 操作系统）。

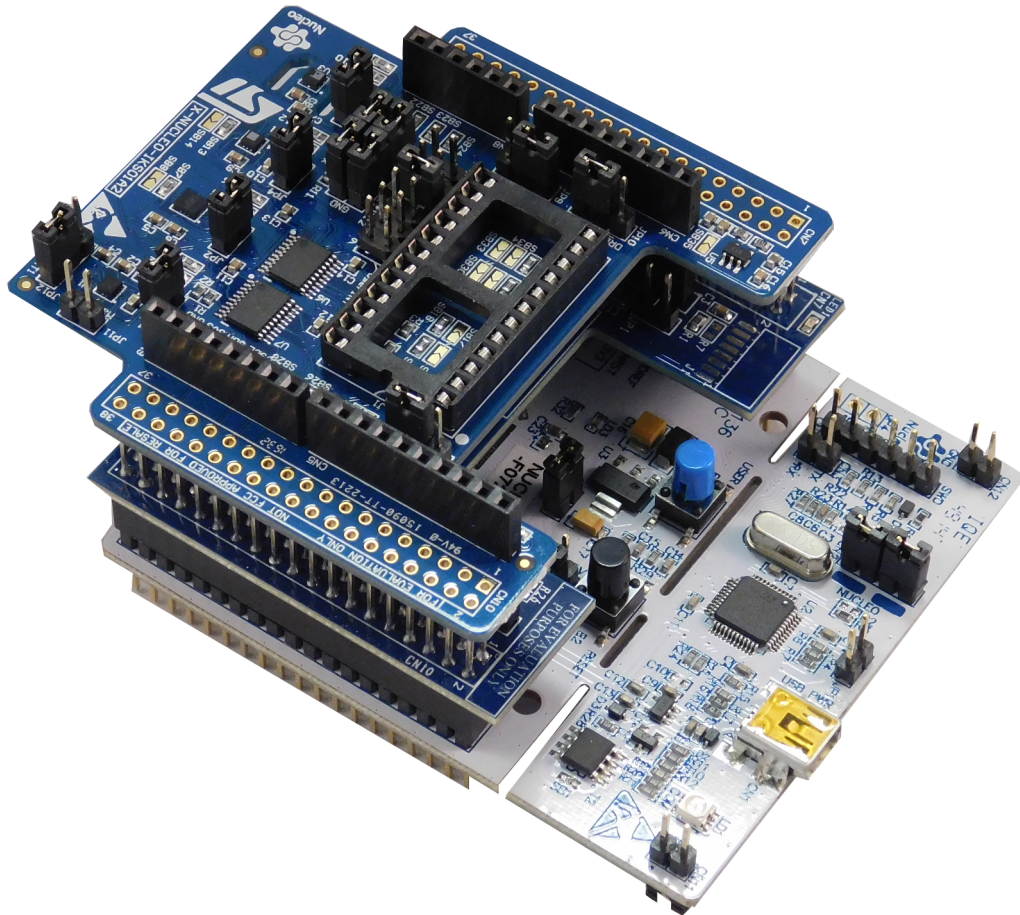
X-NUCLEO-IDW01M1 Wi-Fi 扩展板可通过 morpho 扩展接口连接至 STM32 Nucleo 板，如下所示。

图 14. 通过 morpho 连接器连接至 STM32 Nucleo 板的 X-NUCLEO-IDW01M1 Wi-Fi 扩展板



最后，X-NUCLEO-IKS01A2 或 X-NUCLEO-IKS01A1 传感器扩展板可以很容易地通过 Arduino UNO R3 接口连接至 X-NUCLEO-IDW01M1 Wi-Fi 扩展板，如下所示。

图 15. X-NUCLEO0IKS01A2 传感器扩展板通过叠加 X-NUCLEO-IDW01M1 的顶部来连接到 STM32NUCLEO



5.3 软件要求

为了编译和运行 FP-CLD-AWS1 软件包，需要下述软件组件：

- FP-CLD-AWS1 软件，可从 www.st.com/stm32code 下载
- 以下开发工具链和编译器之一：
 - IAR Embedded Workbench for ARM® (EWARM) 工具链 + ST-LINK
 - RealView Microcontroller Development Kit (MDK-ARM) 工具链 + ST-LINK
 - 用于 STM32 (SW4STM32) 的 System Workbench + ST-LINK
- 调试终端（例如 Tera Term，http://www.compuphase.com/software_termite.htm）

5.4 软件设置

本节为开发人员列出了设置 FP-CLD-AWS1 SDK、运行示例测试场景以及自定义应用的最低要求。

5.4.1 开发工具链和编译器

请选择一个受软件包支持且在第 5.3 节 软件要求中列出的集成开发环境，并阅读所选 IDE 提供商提供的系统和设置信息。支持的全部三种 IDE 的项目文件均可以在 FP-CLD-AWS1 软件包的 Projects/Multi/Applications/MQTT_AWS/文件夹中找到。

版本历史

表 2. 文档版本历史

日期	版本	变更
2017 年 3 月 22 日	1	初始版本

目录

1	缩写和缩略语	2
2	FP-CLD-AWS1 软件包概述	3
2.1	概述	3
2.2	架构	3
2.3	文件夹结构	4
2.4	API	4
3	Amazon AWS IoT 服务设置	6
3.1	概述	6
3.2	AWS 管理控制台配置步骤	6
3.2.1	设备证书	6
3.2.2	使用 OpenSSL 创建 CSR 并上传到 AWS IoT 控制台	6
4	FP-CLD-AWS1 示例应用描述	9
4.1	通过 AWS IoT 配置更新源代码	9
4.1.1	aws_iot_config.h 中的更改	9
4.1.2	aws_nucleo_certificates.c 中的更改	9
4.2	配置 Wi-Fi 接入点参数并将 STM32 Nucleo 板连接到 AWS IoT 控制台	11
5	系统设置指南	14
5.1	硬件说明	14
5.1.1	STM32 Nucleo 平台	14
5.1.2	X-NUCLEO-IDW01M1 扩展板	15
5.1.3	X-NUCLEO-IKS01A1 扩展板	15
5.1.4	X-NUCLEO-IKS01A2 扩展板	16
5.2	硬件设置	17
5.2.1	STM32 Nucleo 和扩展板设置	17
5.3	软件要求	19
5.4	软件设置	19
5.4.1	开发工具链和编译器	19
	版本历史	20

图一览

图 1.	FP-CLD-AWS1 软件架构	4
图 2.	FP- CLD-AWS1 软件包文件夹结构	4
图 3.	在 AWS 控制台中通过 CSR 文件创建证书	7
图 4.	将 CSR 文件上传到 AWS 控制台	7
图 5.	成功创建证书	8
图 6.	AWS IoT 控制面板：设备详细信息	9
图 7.	证书下载	10
图 8.	通过串行接口调试工具设置 SSID 和密码	12
图 9.	终端传感器数据查看	13
图 10.	STM32 Nucleo 板	14
图 11.	X-NUCLEO-IDW01M1 Wi-Fi 扩展板	15
图 12.	X-NUCLEO-IKS01A1 扩展板	16
图 13.	X-NUCLEO-IKS01A2 MEMS 和环境传感器扩展板	17
图 14.	通过 morpho 连接器连接至 STM32 Nucleo 板的 X-NUCLEO-IDW01M1 Wi-Fi 扩展板	18
图 15.	X-NUCLEO0IKS01A2 传感器扩展板通过叠加 X-NUCLEO-IDW01M1 的顶部来连接到 STM32NUCLEO	19

重要通知 - 请仔细阅读

意法半导体公司及其子公司（“ST”）保留随时对 ST 产品和/或本文档进行变更、更正、增强、修改和改进的权利，恕不另行通知。买方在订货之前应获取关于 ST 产品的最新信息。ST 产品的销售依照订单确认时的相关 ST 销售条款。

买方自行负责对 ST 产品的选择和使用，ST 概不承担与应用协助或买方产品设计相关的任何责任。

ST 不对任何知识产权进行任何明示或默示的授权或许可。

转售的 ST 产品如有不同于此处提供的信息的规定，将导致 ST 针对该产品授予的任何保证失效。

ST 和 ST 标志是 ST 的商标。关于 ST 商标的其他信息，请访问 www.st.com/trademarks。其他所有产品或服务名称是其各自所有者的财产。

本文档中的信息取代本文档所有早期版本中提供的信息。

© 2019 STMicroelectronics - 保留所有权利