UM2242
用户手册

STM32Cube的Microsoft® Azure IoT云软件扩展入门

前言

本用户手册介绍了STM32Cube的STM32 Microsoft® Azure IoT（物联网）云软件扩展包的内容。

Microsoft® Azure是由Microsoft®创建的云计算服务，通过Microsoft®全球网络管理数据中心来构建、测试、部署和管理应用和服务。它提供软件即服务（SaaS）、平台即服务（PaaS）和基础架构即服务（IaaS），并支持许多不同的编程语言、工具和框架，包括Microsoft®特定的和第三方的软件和系统。

STM32Cube（X-CUBE-AZURE）的Microsoft® Azure IoT软件扩展包提供了将STMicroelectronics板连接到Azure IoT Hub的应用示例。

X-CUBE-AZURE可运行于B-L475E-IOT01、32F413HDISCOVERY和32F769IDISCOVERY板。

实现示例包括设备到云遥测报告和云到设备消息，可向所连接的设备发送命令和通知。还实现了具有应答确认的消息传送跟踪。

X-CUBE-AZURE具有以下特性：

- 已准备好运行固件示例，使用WiFi®和以太网连接来支持Azure设备应用的快速评估和开发
- 具有配置板子连接到Azure IoT Hub的接口
- 连接到Azure IoT Hub和各种回调注册
- Azure IoT Hub，实现了双向通信示例
- B-L475E-IOT01板可测量和报告以下值：
  - 湿度
  - 温度
  - 3D地磁数据
  - 3D加速度
  - 3D陀螺仪数据
  - 大气压力
  - 接近

目录

表格索引

图片索引

1 缩略语

*表 1*给出了相关的缩略语定义，帮助您更好地理解本文档。

表1. 缩略语列表

术语	定义
API	应用编程接口
BSP	板级支持包
C2D	云到设备
CA	认证机构
D2C	设备到云
DHCP	动态主机配置协议
DNS	域名服务器
HAL	硬件抽象层
IDE	集成开发环境
IoT	物联网
IP	互联网协议
JSON	JavaScript对象符号
LED	发光二极管
RTC	实时时钟
UART	通用异步收发器

2 Azure IoT Hub

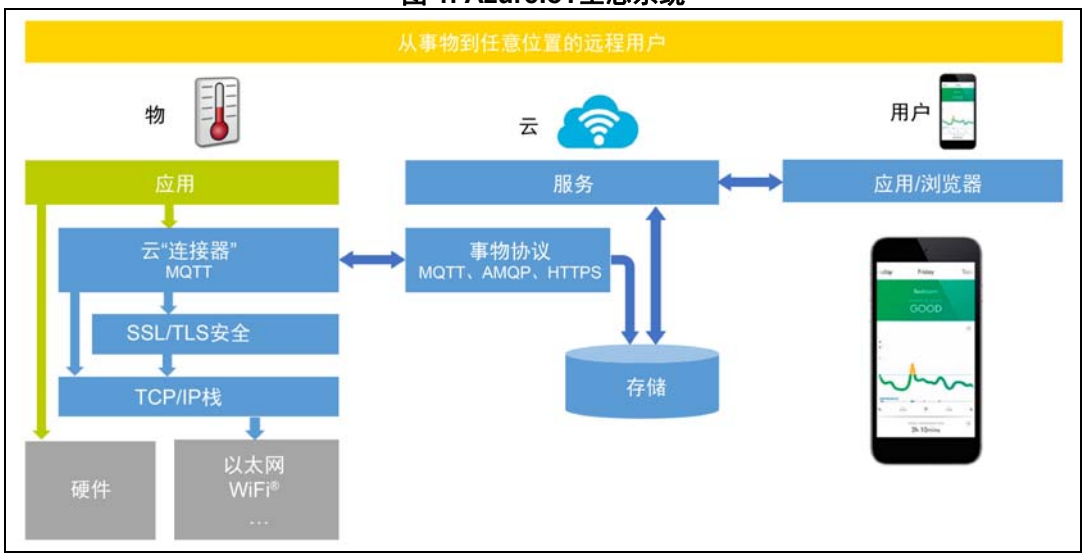
本节介绍Azure IoT Hub服务。

X-CUBE-AZURE包实现了使用C语言的Azure IoT设备SDK，可支持该板安全地连接到Azure IoT Hub服务。

用户可以使用智能手机或个人计算机连接到云端，并可以随时随地访问该板提供的信息。

*图 1*显示面向X-CUBE-AZURE包的Azure IoT生态系统。X-CUBE-AZURE包采用MQTT协议实现物与Azure IoT Hub服务的连接。应用/浏览器和其他物协议在*图 1*中仅显示了其作为现有Azure功能的信息。

图 1. AzureIoT生态系统



除了丰富的设备到云（D2C）和云到设备（C2D）通信选项（包括消息、文件传输和请求-回复方法）之外，Azure IoT Hub还通过以下方式实现设备连接：

- 设备孪生。使用设备孪生，用户可以存储、同步和查询设备元数据和状态信息。设备孪生是存储设备状态信息（元数据、配置和条件）的JSON文档。对于连接到IoT Hub的每个设备，IoT Hub都会保持其设备孪生。
- 每个设备的认证和安全连接。用户可以为每个设备提供自己的安全密钥，使其能够连接到IoT Hub。IoT Hub身份注册表将设备身份和密钥存储在解决方案中。解决方案后端可以将个人设备添加到允许/拒绝列表，以实现对设备访问的完全控制。
- 基于声明规则，将设备到云消息路由到Azure服务。IoT Hub允许用户根据路由规则定义消息路由，以控制中心发送设备到云消息的位置。路由规则不需要用户编写任何代码，并可以取代自定义后置消息调度程序。
- 监控设备连接操作。用户可以收到有关设备身份管理操作和设备连接事件的详细操作日志。这种监控功能使其IoT解决方案能够识别连接问题，例如尝试连接证书错误的设备，太频繁地发送消息或拒绝所有云到设备消息。
- 大量的设备库Azure IoT设备SDK可用于并支持多种语言和平台：C语言可适用于多种Linux®发行版本、Windows®和实时操作系统。Azure IoT设备SDK还支持管理诸如C＃、Java和JavaScript等语言。
- IoT协议和可扩展性。如果某解决方案无法使用设备库，则IoT Hub会公开一种公共协议，使设备本身能够使用MQTTv3.1.1、HTTP 1.1或AMQP 1.0协议。用户还可以通过以下方式扩展IoT Hub以支持自定义协议：
  - 使用AzureIoT Edge创建现场网关，将自定义协议转换为IoT Hub可理解的三种协议之一。
  - 自定义Azure IoT协议网关，即在云中运行的开源组件。
- 扩展。Azure IoT Hub可扩展到数百万个同时连接的设备和每秒数百万的事件。

有关Microsoft® Azure和Azure IoT Hub的完整说明，请参考Azure IoT Hub服务网页概述中提供的信息。

3 包说明

本节详细介绍了X-CUBE-AZURE包的内容和使用方法。

3.1 通用描述

X-CUBE-AZURE包为STM32微控制器提供了Azure堆栈中间件。

已经移植到了B-L475E-IOT01、32F413HDISCOVERY和32F769IDISCOVERY板，并可通过板载的网络接口连接到互联网。

- B-L475E-IOT01支持WiFi®与板上Inventek模块的连接。该板配备了一组能够报告湿度、温度、3D地磁数据、3D加速度、3D陀螺仪数据、大气压力、接近度和手势检测（X-CUBE-AZURE不使用手势检测功能）的传感器。
- 32F413HDISCOVERY支持WiFi®与板上Inventek模块的连接。
- 32F769IDISCOVERY本身提供了一个以太网接口。

该包分为以下组件：

- C99 SDK，用来将设备连接到Microsoft® Azure IoT服务
- mbedTLS
- LwIP
- FreeRTOS™
- WiFi®驱动
- 用于32F769IDISCOVERY板的以太网驱动
- 用于B-L475E-IOT01板的传感器驱动
- STM32L4系列、STM32F4系列，和STM32F7系列HAL
- Azure应用程序示例

该软件以zip文档的形式提供，其中包含源代码。

可支持以下集成开发环境：

- 用于ARM®（EWARM）的IAREmbeddedWorkbench®。必须使用7.80.4或更高的IAR™版本7
- Keil®微控制器开发套件（MDK-ARM）
- 用于STM32的System Workbench。必须使用1.14.0或更高版本

3.2 架构

本节描述X-CUBE-AZURE包的软件组成部分。

X-CUBE-AZURE软件是对STM32Cube的扩展。其主要功能和特性如下：

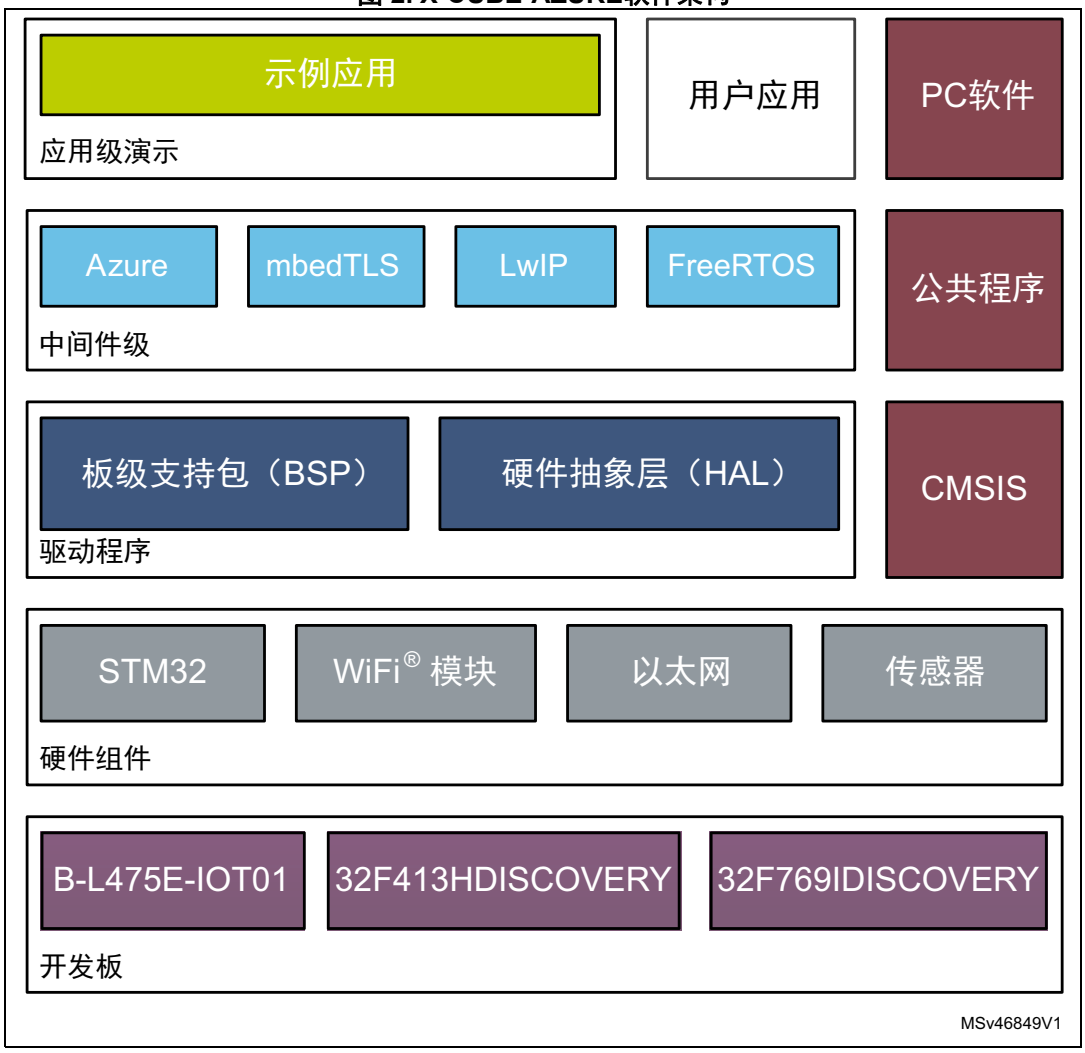
- 完全兼容STM32Cube架构
- 为了能够开发应用程序来访问和使用Azure IoT，扩展了STM32Cube。
- 基于STM32CubeHAL，它是STM32微控制器的硬件抽象层

应用软件访问和使用Azure IoT Hub所用的软件组件如下：

- STM32Cube HAL
  HAL驱动层提供通用的多实例简单API（应用程序编程接口），以便与上层（应用、库和堆栈）交互。
  它由通用和扩展API构成。它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，实现了它的功能又无需依赖给定微控制器单元（MCU）的特定硬件配置。
  此结构可提高库代码的可复用性，并确保可向其他设备轻松移植。
- 板级支持包（BSP）
  除MCU之外，软件包需支持STM32板上的外设。板级支持包（BSP）中包含此软件。这是一个有限的API集，为板特有的某些外设（例如LED和用户按钮等）提供编程接口。
- Azure中间件
  它由Azure IoT Hub客户端库、JSON解析器、JSON序列化程序、MQTT客户端（由IoT Hub客户端库用作传输层）以及客户端库使用的各种C公共程序组成。
- mbedTLS
  Azure中间件使用由mbedTLS库管理的TLS连接。
- TCP/IP
  TCP/IP连接可由WiFi®模块（当使用WiFi®连接时）或LwIP中间件（当使用以太网连接时）来处理。在X-CUBE-AZURE包中，只有32F769IDISCOVERY板可以通过以太网进行连接。
- FreeRTOS™
  它是一个实时操作系统，LwIP要求为用户提供基于套接字的接口。

*图 2*概括了X-CUBE-AZURE软件架构。

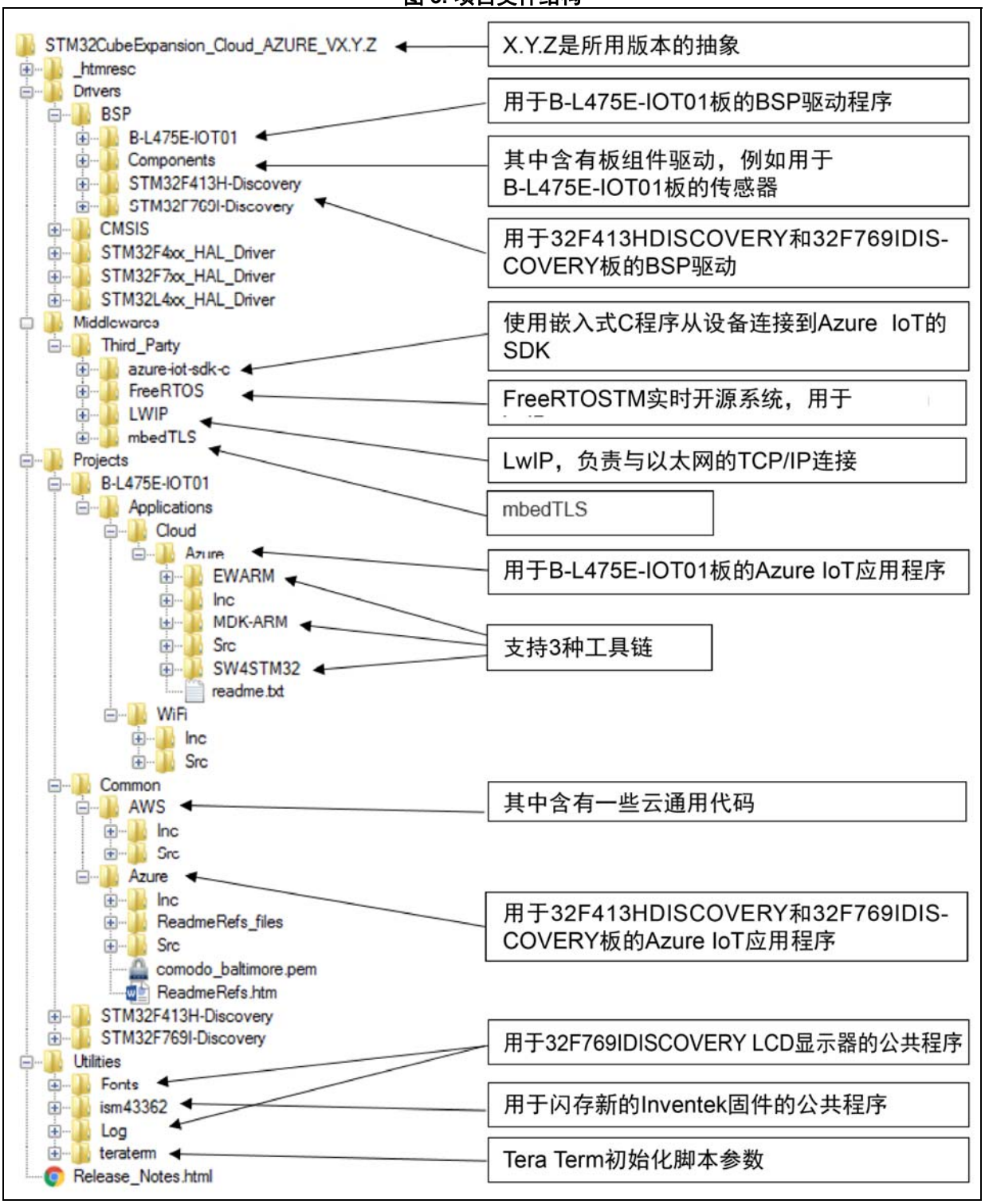
图 2. X-CUBE-AZURE软件架构



3.3 文件夹结构

*图 3*给出了X-CUBE-AZURE包的文件夹结构。

图 3. 项目文件结构



3.4 B-L475E-IOT01板传感器

板上由示例应用使用的传感器是：

- 用于相对湿度和温度测量的电容式数字传感器（HTS221）
- 高性能3轴磁力计（LIS3MDL）
- 3D加速度计和3D陀螺仪（LSM6DSL）
- 260-1260 hPa绝对数字输出气压计（LPS22HB）
- 接近传感器（VL53L0X）

发布传感器信息的示例：

```
{
 "mac": "<mac address of the device>",
 "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,
 "ts": "2017-06-07T15:14:22Z"
}
```

*表 2*给出了B-L475E-IOT01板传感器报告值的单位。

表2. B-L475E-IOT01板传感器报告值的单位

数据	单位
温度	摄氏度（°C）
湿度	相对湿度（%）
压力	百帕斯卡（hPa）
接近	毫米（mm）
加速度	毫重力（mgforce）
角速度	毫度每秒（mdps）
磁感应	毫高斯（mG）

3.5 WiFi®组件

WiFi®软件分为用于模块专用软件的驱动程序/BSP/组件，以及用于I/O操作和WiFi®模块抽象的Projects/<board>/WiFi。

3.6 复位按钮

复位按钮（黑色）用于随时复位板子。此操作可使板重新启动。

3.7 用户按钮

用户按钮（蓝色）用于以下情况：

- 要配置WiFi®和Azure安全证书时。从板启动时开始直到启动后五秒，都可以完成此操作。
- 当初始化板来控制数据发布到Azure IoT Hub的方式时，请参见*图 7*

应用程序通过板级支持包（BSP）功能来配置和管理用户按钮。

BSP功能位于Drivers\BSP\<board name>目录中。

使用BSP按钮功能时（采用BUTTON_USER值），对于给定的平台，应用程序不会从硬件角度考虑此按钮连接的方式。该映射由BSP处理。

3.8 用户LED

应用程序所用的用户LED配置通过板级支持包（BSP）功能完成。

BSP功能位于Drivers\BSP\<board name>目录下。

使用BSP按钮功能时（采用LED_GREEN值），对于给定的平台，应用程序不会考虑LED映射的方式。该映射由BSP处理。

3.9 实时时钟

STM32 RTC在启动时从www.gandi.net Web服务器进行更新。

用户可以使用HAL_RTC_GetTime()函数得到时间值。

例如，此函数可以用于时间戳消息。

3.10 mbedTLS配置

利用 #include配置文件，mbedTLS中间件支持是完全可配置的。

利用MBEDTLS_CONFIG_FILE #define，可以覆盖配置文件名。

X-CUBE-AZURE包使用文件az_mbedtls_config.h来实现项目配置。

这可以通过在mbedTLS.c和mbedTLS.h文件开头使用以下# 指令来实现：

```
#if !defined(MBEDTLS_CONFIG_FILE)
#include "mbedtls/config.h"
#else
#include MBEDTLS_CONFIG_FILE
#endif
```

配置文件指定了要集成的密码。

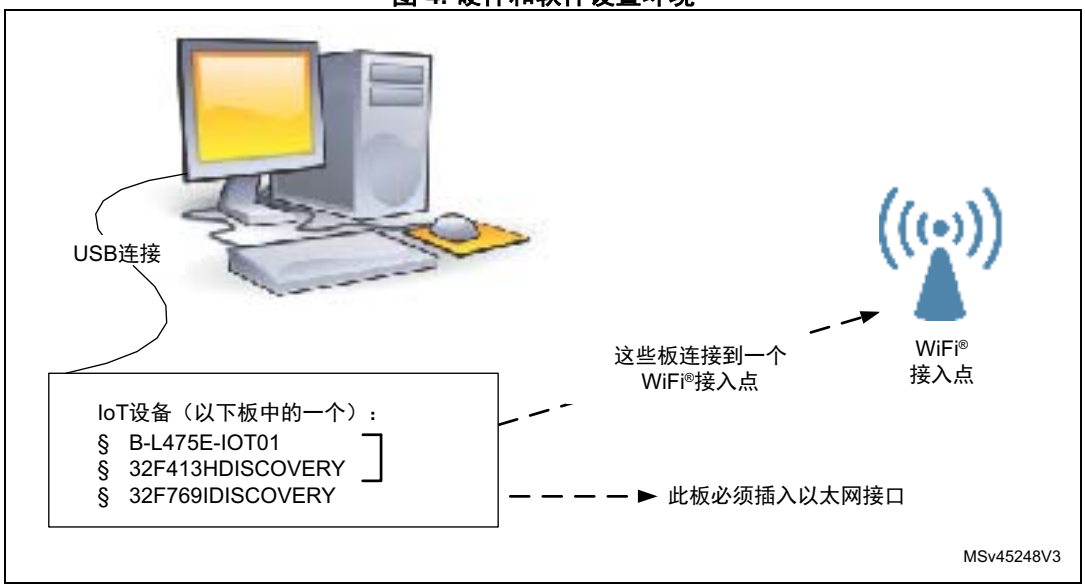
4 硬件和软件环境设置

要设置硬件和软件环境，必须利用USB线缆将某种支持的板（共3种）插入个人计算机。与PC的连接允许用户：

- 烧写板
- 存储WiFi®和Azure安全证书。
- 通过UART控制台与板进行交互
- 调试

B-L475E-IOT01或32F413HDISCOVERY板必须连接到WiFi®接入点，而32F769IDISCOVERY
板必须连接到以太网接口，如*图 4*中所示。

图 4. 硬件和软件设置环境



运行示例的先决条件是：

- 一个WiFi®接入点，具有透明的互联网连接，这意味着代理服务器和防火墙都不能阻止输出流量。它必须运行一个DHCP服务器，将IP和DNS配置提供给主板。
- 一台用于运行设备管理应用程序的计算机，具有透明的互联网连接，这意味着代理服务器和防火墙都不能阻止输出流量。例如，它可以是开发PC、虚拟专用服务器或单板计算机。它可以连接到与MCU板相同的路由器。

- 一个Azure IoT账户，用来创建IoT Hub。在运行实例之前请参考GitHub网页上的iot-hub-device-sdk-c-intro
- 与设备进行通信所需的Azure设备管理应用程序。

  有两个选项可以从上述网页下载。

  本文档中，我们选择了iothub-explorer工具作为本Cube扩展包的参考。它可以在任意操作系统上运行。它是一个node.js应用程序。

  或者，可以使用Windows®（.NET）上运行的设备资源管理器
- 一台开发PC，用于构建应用程序、利用ST-Link进行编程，以及运行虚拟控制台。

5 与板交互

需要串行终端来：

- 配置板
- 显示本地接收到的Azure IoT C2D消息

使用Tera Term对本文档中的示例进行说明。也可以使用其他类似的工具。

当板首次使用时，必须使用Azure IoT标识数据进行编程。

- 确定PC上所用的用于探索板的STM32 ST-LINK虚拟COM端口。在一台Windows® PC上，打开设备管理器
- 打开PC上的虚拟终端并将其连接到上述虚拟COM端口。

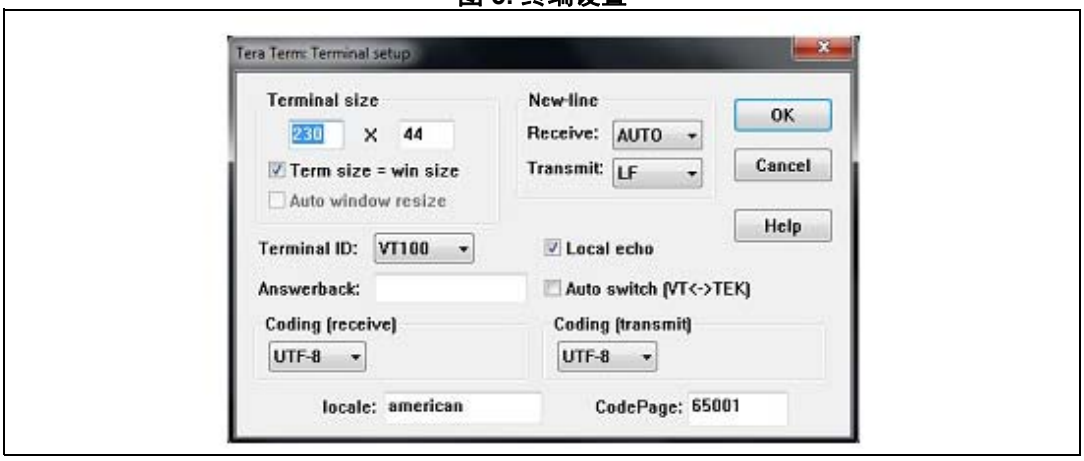
在开发包实用程序目录中提供了一个Tera Term初始化脚本（参考*图 3*）；此脚本设置了正确的参数。要使用它，请打开Tera Term，选择Setup，然后选择Restore设置。

注： *本章中下面提供的信息可用来配置UART终端，使其成为Tera Term初始化脚本的替代选择。*

终端设置如*图 5*中所示，其中显示了终端设置和新行的建议参数。

虚拟终端新行传输配置必须设置为LineFeed（\n或LF），以便允许从UNIX类型的文本文件复制粘贴。本地回显选项使得复制粘贴在控制台上可见。

图 5. 终端设置

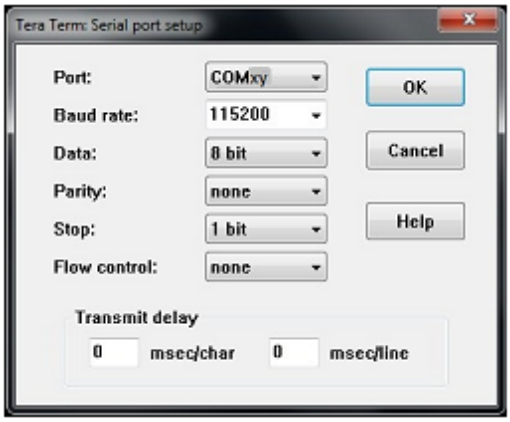


串口必须配置：

- COM端口编号
- 115200波特率
- 8位数据
- 无校验位
- 1个停止位
- 无流控

串口设置如*图 6*中所示。

图 6. 串口设置



UART终端和串口设置完毕后，按下板复位按钮（黑色）。

按照UART终端上的指示上传WiFi®和Azure数据。这些数据保留在Flash中，并在下次板启动时重复使用。

6 应用程序示例

6.1 应用描述

AzureXcubeSample应用程序说明了Azure设备与Azure IoT Hub进行交互的各种方式。

基于用户在控制台上提供的证书，该应用程序可连接到Azure IoT Hub上。

6.2 应用设置

应用程序设置需要按照从*第 6.2.1 节*到*第 6.2.4 节*所描述的步骤依次执行。

6.2.1 Azure设备创建

下面两个命令可以创建一个Azure设备并获取其连接字符串：

- $ iothub-explorer login <your Azure IoT Hub Connection String>
- $ iothub-explorer create <devId> --connection-string

板MAC地址可以用作设备ID。为了方便也可以选择一个巧妙的昵称。

建议保留设备连接字符串的副本，因为首次启动时AzureXcubeSample应用程序会在控制台上请求该字符串。

下一个命令用来验证设备孪生是否恢复：

- $ iothub-explorer get-twin <devId>

6.2.2 应用程序构建和烧写

在STM32CubeExpansion_Cloud_AZURE_Vx.y.z\Projects\<board name>\Applications\Cloud\Azure\<IDE>下打开所选工具链并构建项目。

有关IDE版本要求的详细信息，请参阅*第 3.1 节：通用描述 第 8页*。

6.2.3 STM32板上的固件编程

将STM32CubeExpansion_Cloud_AZURE_Vx.y.z\Projects \<board name>\Applications\Cloud\Azure\<IDE>\Exe中生成的二进制文件复制或拖放到STM32板插入PC时所创建的USB大容量存储位置。

如果主机是一台Linux® PC，则可以在/media文件夹中找到名为DIS_L4IOT的STM32设备。例如，如果创建的大容量存储位置为/media/DIS_L4IOT，那么使用名为my_firmware.bin的二进制文件对该板进行编程的命令是很简单的：

cp my_firmware.bin /media/DIS_L4IOT。

或者，可以利用支持的某一开发工具链，直接对STM32板进行编程。

6.2.4 应用程序首次启动

必须通过USB（ST-LINK USB端口）将板连接到PC。

通过串行终端模拟器（如Tera Term）打开控制台（参考*第 3.2节：架构 第 9页*）。

在控制台上：

- 对于支持WiFi®的板，请输入Wifi® SSID、加密模式和密码
- 设置设备连接字符串（参考*第 6.2.1节*），不包括引号（"）
- 通过复制粘贴STM32CubeExpansion_Cloud_AZURE_Vx.y.z\Projects\Common\Azure\comodo_baltimore.pem的内容，来设置TLS根CA证书。
  该设备使用它们通过TLS对远程主机进行身份验证。

注： *AzureXcubeSample应用程序要求提供2个CA证书的连结*

1. 用于HTTPS服务器，它用来在启动时检索当前时间和日期。（www.gandi.net服务器的Comodo证书）

2. 用于IoT Hub服务器（Baltimore证书）

串接的字符串必须以空行结尾。这是comodo_baltimore.pem。

参数配置完成后，可以通过在启动后重启板并按下用户按钮（蓝色按钮）来进行更改。

6.3 应用程序运行时

本节介绍应用程序的生命周期步骤：

- 进行单个HTTPS请求以检索当前时间和日期，并配置RTC
- 连接到Azure IoT Hub
- 获取设备孪生的状态
- 根据设备孪生所需的属性来更新其本地属性（DesiredTelemetryInterval）
- 将报告属性报告到设备孪生（TelemetryInterval和LedStatus）
  注意：根据这一点，用户可以通过
  $ iothub-explorer get-twin <devId>命令获取所更新的孪生状态
- 保持空闲状态，等待本地用户或中心发起的事件

从这点出发，可能的本地用户操作是：

- 单击用户按钮：此操作通过DeviceToCloud（D2C）消息触发对IoT Hub的消息发布。
- 双击用户按钮：此操作启动或停止消息发布循环。该循环运行时，每隔TelemetryInterval秒都会发布消息。

  注意：每次消息发布都是以用户LED快速闪烁半秒钟的方式发出信号。

  消息内容取决于所用板的类型：

  – *B-L475E-IOT01报告传感器值和时间戳*

  – *32F413HDISCOVERY和32F769IDISCOVERY只报告时间戳*

所执行的中心发起事件是：

- CloudToDevice（C2D）消息
  该消息会显示在板控制台上
- C2D孪生更新
  用来更改遥测发布周期（即DesiredTelemetryInterval参数）
- C2D方法
  用来调用以下某一设备方法：
  – Reboot：重新启动板
  – Hello：在板控制台上显示作为参数传递的消息
- C2D操作调用
  可调用LedToggle来实现用户LED状态切换

*图 7*显示了运行时状态流图。

图 7. 运行时状态流图

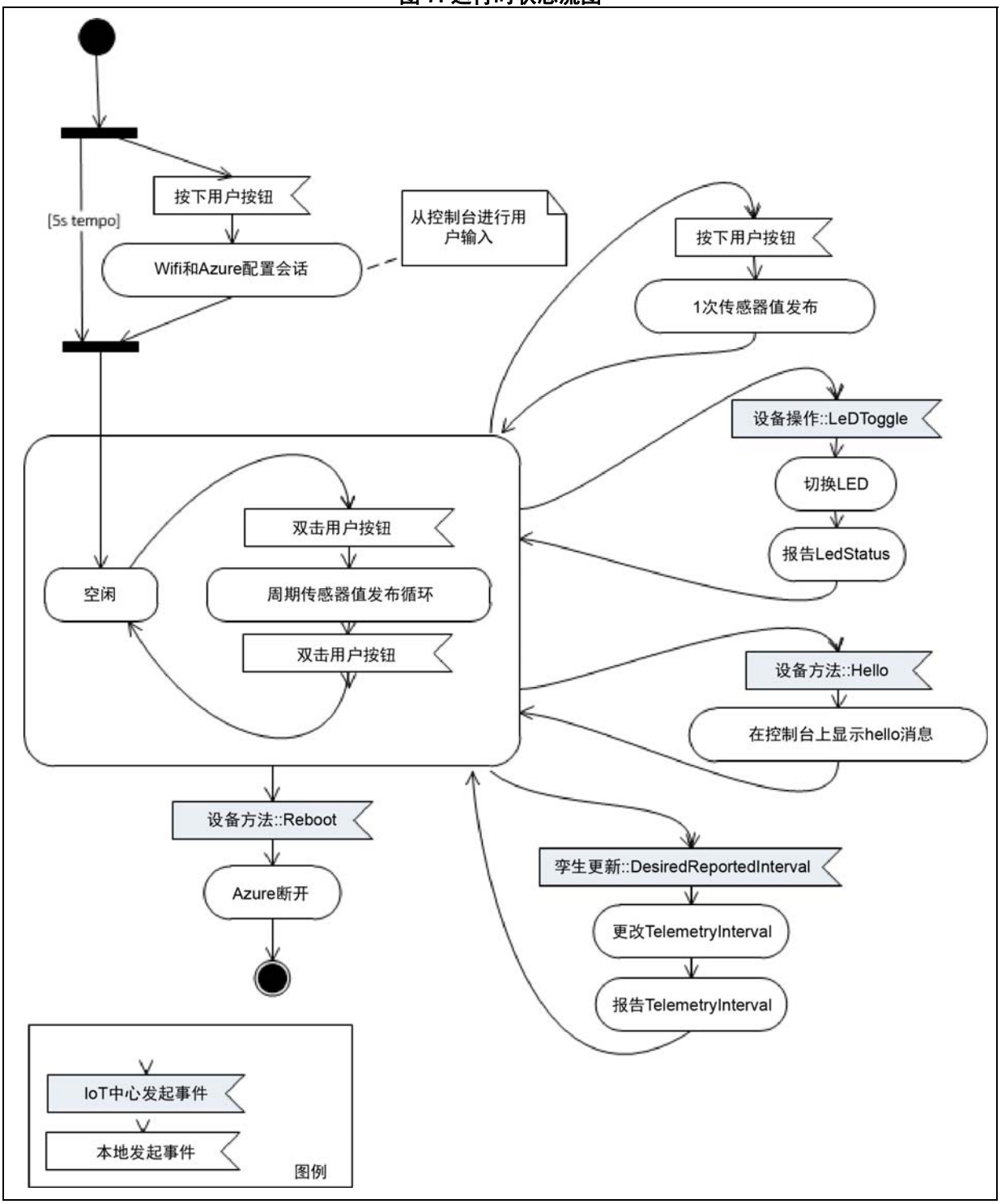

[5s tempo]
按下用户按钮
Wifi和Azure配置会话
从控制台进行用户输入
空闲
双击用户按钮
周期传感器值发布循环
双击用户按钮
设备方法::Reboot
Azure断开
按下用户按钮
1次传感器值发布
设备操作::LeDToggle
切换LED
报告LedStatus
设备方法::Hello
在控制台上显示hello消息
孪生更新::DesiredReportedInterval
更改TelemetryInterval
报告TelemetryInterval
IoT中心发起事件
本地发起事件
图例

*表 3*列出了iothub-explorer命令行，以供用户触发中心发起的事件，并查看结果。列出了设备和云之间的七个通信接口。使用iothub-explorer节点包来按列调用命令。

表3. iothub-explorer命令行

通信接口	应用程序用途	调用方式，或格式	备注

D2C消息	发布遥测数据	monitor-events -l <IoTHubConnectionString> <devId>
		>
		{
		"mac": "<mac address of the device>",
		"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,
		"ts": "2017-06-07T15:14:22Z"
		}
			如果用户利用板上的用户按钮激活了发布，则可以监控遥测信息。请参见*图 7*。

C2D消息	发送Hello world	send <devId> 'Hello world'	AzureSDK打印错误日志，因为设备上的C2D JSON命令解析需要实现消息回调，同时发送简单的文本。

表3. iothub-explorer命令行（续）

通信接口	应用程序用途	调用方式，或格式	备注
C2D孪生更新	改变遥测发布周期	update-twin <devId> '{ "properties": { "desired": { "DesiredTelemetryInterval": 6 } } }'	客户端SDK、iothub-explorer和中心为孪生更新提供不同的有效载荷格式： 连接时，从中心接收完整的字符串：{ "desired": { "DesiredTelemetryInterval": x } } 运行时，当使用更新孪生，仅包含 { "DesiredTelemetryInterval": x } 设备上的回调实现与两种格式均可兼容。 它仍然调用JSON解析器来识别格式。 如果设置了AzureSDK logtrace选项，则在找不到所需密钥时会打印错误日志。 在孪生更新中，新的TelemetryInterval会通过D2C孪生更新自动报告。

D2C孪生更新	报告遥测激活参数和LED状态	孪生更新由设备上运行的应用程序自动执行。
		可按以下步骤恢复更改：
		get-twin [-r] <devId>
		>
		{
		"deviceId":"C47F510111A7",
		"properties":{
		"desired":{
		"DesiredTelemetryInterval":6,
		},
		"reported":{
		"TelemetryInterval":6,
		"LedStatusOn":false
		}
		}
		}
			在连接时、C2D孪生更新时和C2D LedToggle调用时进行更新。

C2D方法	重新启动，发送消息	device-method <devId> Reboot '{ "when": "now"}' an device-method <devId> Hello '{ "msg": "World!" }'	为了实现互操作性，必须将有效的JSON字符串作为参数传递给每个设备方法调用，即使目标函数没有参数。 Reboot实现不使用任何传递参数。

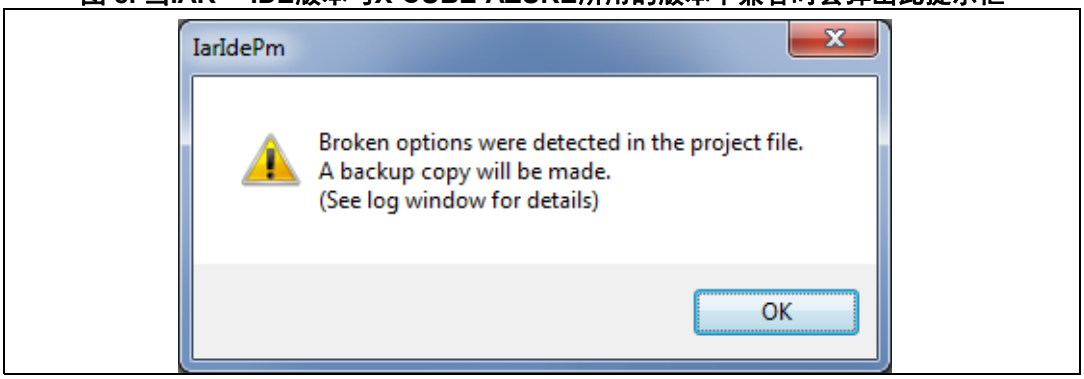
表3. iothub-explorer命令行（续）

通信接口	应用程序用途	调用方式，或格式	备注
C2D调用	发送LedToggle消息以便设备解析	send <devId> '{ "Name": "LedToggle", "Parameters": "" }'	为了实现互操作性，必须赋予Parameters键一个值（任何值）。
D2C上传到二进制大对象	未使用	-	未实现。

7 常见问题

问：当我用IAR™打开项目时，为什么会弹出此提示框（见*图 8*）？

图 8. 当IAR™ IDE版本与X-CUBE-AZURE所用的版本不兼容时会弹出此提示框



答：很可能是由于此IAR™IDE版本比开发包所用版本更早（参见*第 3.1 节：通用描述 第 8 页*），因此不能保证兼容性。这种情况下，需要更新IAR™ IDE版本。

问：我的设备不能连接到WiFi®接入点。我该如何处理？

答：请确认其他设备是否可以连接到WiFi®接入点。如果可以，请在板重置后，按下用户按钮（蓝色）五秒钟，输入WiFi®凭证。

问：接近传感器总是报告“8190”，即使我将障碍物靠近其放置也是如此。

A：请确保已经取下衬垫（放置在接近传感器上的非常薄的薄膜）。它的颜色是橙色，不是很明显。

8 版本历史

表4. 文档版本历史

日期	版本	变更
2017年7月20日	1	初始版本。

表5. 中文文档版本历史

日期	版本	变更
2017年10月27日	1	中文初始版本。

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 徽标是 ST 的商标。所有其他产品或服务名称均为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。本文档的中文版本为英文版本的翻译件，仅供参考之用；若中文版本与英文版本有任何冲突或不一致，则以英文版本为准。