|  |
| --- |
| OpenCPU |
| SDK MQTT API说明 |
| 版本1.1日期2025-02-26 |

版权声明

版权所有 © 深圳市有方科技股份有限公司2025。深圳市有方科技股份有限公司保留所有权利。

未经深圳市有方科技股份有限公司书面同意，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

是深圳市有方科技股份有限公司所有商标。

本文档中出现的其他商标，由商标所有者所有。

说明

本文档的使用对象为系统工程师，开发工程师及测试工程师。

本设计指南为客户产品设计提供支持，客户须按照本文中的规范和参数进行产品设计和调试。如因客户操作不当造成的人身伤害和财产损失，有方概不承担责任。

由于产品版本升级或其它原因，本文档内容会在不预先通知的情况下进行必要的更新。

除非另有约定，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

深圳市有方科技股份有限公司为客户提供全方位的技术支持，任何垂询请直接联系您的客户经理或发送邮件至以下邮箱：

Sales@neoway.com

Support@neoway.com

公司网址：http://www.neoway.com

目录

[关于本文档 iv](#_Toc191470907)

[范围 iv](#_Toc191470908)

[读者对象 iv](#_Toc191470909)

[修订记录 iv](#_Toc191470910)

[符号约定 iv](#_Toc191470911)

[1 概述 5](#_Toc191470912)

[2 API 介绍 6](#_Toc191470913)

[2.1 数据结构 6](#_Toc191470914)

[2.1.1 MQTTClient 客户端实例 6](#_Toc191470915)

[2.1.2 void (\*messageHandler)(MessageData\*)消息处理回调 6](#_Toc191470916)

[2.1.3 MQTTMessage消息处理回调参数 6](#_Toc191470917)

[2.1.4 Network封装和管理网络通信的核心数据和操作接口 7](#_Toc191470918)

[2.1.5 TLSConnectParams TLS连接参数结构体 7](#_Toc191470919)

[2.2 MQTTIsConnected 7](#_Toc191470920)

[2.3 NetworkInit 8](#_Toc191470921)

[2.4 MQTTClientInit 8](#_Toc191470922)

[2.5 NetworkConnect 8](#_Toc191470923)

[2.6 MQTTConnect 8](#_Toc191470924)

[2.7 MQTTPublish 9](#_Toc191470925)

[2.8 MQTTSubscribe 9](#_Toc191470926)

[2.9 MQTTSubscribe 9](#_Toc191470927)

[2.10 MQTTDisconnect 10](#_Toc191470928)

[2.11 NetworkDisconnect 10](#_Toc191470929)

[2.12 MQTTYield 10](#_Toc191470930)

关于本文档

范围

本文档对应产品为N706B系列（RTOS系统）。

读者对象

本文档的使用对象为系统工程师，开发工程师及测试工程师。

修订记录

|  |  |  |  |
| --- | --- | --- | --- |
| 版本 | 日期 | 变更 | 作者 |
| 1.0 | 2025-01 | 初始版本 | Niu Sulong |
| 1.1 | 2025-02 | 更新了MQTT相关结构体参数的说明 | Niu Sulong |

符号约定

|  |  |
| --- | --- |
| 符号 | 含义 |
|  | 危险或警告，用户必须遵从的规则，否则会造成模块或客户设备不可逆的故障损坏，甚至可能造成人员身体伤害。 |
|  | 注意，警示用户使用模块时应该特别注意的地方，如不遵从，模块或客户设备可能出现故障。 |
|  | 说明或提示，提供模块使用的意见或建议。 |

# 概述

本文提供基于开源库paho.mqtt.embedded-c-master的api，用于实现MQTT的连接，以帮助客户的开发人员尽快完成相关应用的开发。

# API 介绍

如下接口函数定义位于MQTTClient.h，及MQTTFreeRTOS.h。

## 数据结构

### MQTTClient 客户端实例

typedef struct MQTTClient

{

unsigned int next\_packetid, // 下一个消息包ID

command\_timeout\_ms; // 命令超时时间（毫秒）

size\_t buf\_size, // 发送缓冲区大小（字节）

readbuf\_size; // 接收缓冲区大小（字节）

unsigned char \*buf, // 发送缓冲区指针

\*readbuf; // 接收缓冲区指针

unsigned int keepAliveInterval; // 心跳间隔时间（秒）

char ping\_outstanding; // 是否已发送PING请求但未收到响应（0:否, 1:是）

int isconnected; // 是否已连接到MQTT服务器（0:未连接, 1:已连接）

int cleansession; // 是否使用清理会话模式（0:否, 1:是）

struct MessageHandlers

{

const char\* topicFilter; // 主题过滤器，匹配订阅的主题

void (\*fp) (MessageData\*); // 消息处理函数指针

} messageHandlers[MAX\_MESSAGE\_HANDLERS]; // 消息处理器数组，按订阅主题索引

void (\*defaultMessageHandler) (MessageData\*); // 默认消息处理函数指针

Network\* ipstack; // 网络层接口指针

Timer last\_sent, // 最后一次发送数据的时间

last\_received, // 最后一次接收数据的时间

pingresp\_timer; // PING响应超时时间

#if defined(MQTT\_TASK)

Mutex mutex; // 互斥锁，用于多线程同步

void \* thread\_hdl; // 线程句柄，管理MQTT任务线程

#endif

int active\_close; // 是否主动关闭连接（0:否, 1:是）

} MQTTClient;

### void (\*messageHandler)(MessageData\*)消息处理回调

typedef void (\*messageHandler)(MessageData\*); // 定义消息处理函数指针类型，用于处理接收到的MQTT消息

### MQTTMessage消息处理回调参数

typedef struct MQTTMessage

{

enum QoS qos; // 消息的服务质量等级（QoS 0, 1, 或 2）

unsigned char retained; // 是否为保留消息（0:否, 1:是）

unsigned char dup; // 是否为重复消息（0:否, 1:是）

unsigned short id; // 消息ID，用于标识消息（仅在QoS 1或2时有效）

void \*payload; // 消息负载数据指针，指向实际的消息内容

size\_t payloadlen; // 消息负载长度（字节），表示消息内容的大小

void \* mqtt\_hal; // MQTT当前客户端句柄，多路mqtt时用于标识收到数据的客户端，（可选扩展）

} MQTTMessage;

### Network封装和管理网络通信的核心数据和操作接口

struct Network

{

int cid; // 连接ID，用于标识网络连接

xSocket\_t my\_socket; // 网络套接字，表示底层的网络连接句柄

int (\*mqttread) (Network\*, unsigned char\*, int, int); // MQTT读取函数指针，用于从网络读取数据

int (\*mqttwrite) (Network\*, unsigned char\*, int, int); // MQTT写入函数指针，用于向网络发送数据

void (\*disconnect) (Network\*); // 断开连接函数指针，用于关闭网络连接

boolean is\_SSL; // 是否使用SSL/TLS加密（true: 是, false: 否）

TLSConnectParams tlsConnectParams; // TLS连接参数结构体，包含通用的连接参数（如证书、密钥等）

TLSDataParams tlsDataParams; // TLS数据参数结构体，管理TLS会话和上下文

};

### TLSConnectParams TLS连接参数结构体

typedef struct {

char \*ca\_cert; // 根CA证书内容（字符串形式）

char \*client\_cert; // 客户端证书内容（字符串形式）

char \*client\_key; // 客户端私钥内容（字符串形式）

char \*pRootCALocation; // 指向包含根CA文件路径和文件名的字符串

char \*pDeviceCertLocation; // 指向包含设备证书路径和文件名的字符串

char \*pDevicePrivateKeyLocation; // 指向包含设备私钥文件路径和文件名的字符串

char \*pDestinationURL; // 指向包含MQTT服务端点地址的字符串

uint16\_t DestinationPort; // 定义MQTT服务连接端口的整数

uint32\_t timeout\_ms; // 定义TLS握手超时时间的无符号整数（单位：毫秒）

boolean ServerVerificationFlag; // 布尔值。True = 执行服务器证书主机名验证；False = 跳过验证（不推荐）

int ssl\_version; // 定义使用的SSL/TLS协议版本（如TLS 1.2或TLS 1.3）

} TLSConnectParams;

**注意**：关于证书数据内容和路径参数，2选一即可，可从文件系统中读取证书文件，也可直接使用内容，优先使用证书内容，证书内容为空时会尝试从提供的路径中读取

## MQTTIsConnected

|  |  |
| --- | --- |
| 格式 | int MQTTIsConnected(MQTTClient\* client) |
| 描述 | 获取 MQTT 连接状态 |
| 参数说明 | client：客户端实例 |
| 返回值 | 成功：0  失败：其他 |
| 注意事项 | 无。 |

## NetworkInit

|  |  |
| --- | --- |
| 格式 | void NetworkInit(Network\* n) |
| 描述 | 网络初始化 |
| 参数说明 | n：网络连接参数 |
| 返回值 | 无 |
| 注意事项 | 无。 |

## MQTTClientInit

|  |  |
| --- | --- |
| 格式 | void MQTTClientInit(MQTTClient\* c, Network\* network, unsigned int  command\_timeout\_ms,unsigned char\* sendbuf, size\_t sendbuf\_size,  unsigned char\* readbuf, size\_t readbuf\_size) |
| 描述 | 初始化 MQTT 实例。 |
| 参数说明 | c: MQTT 实例  network: 网络连接参数  command\_timeout\_ms:超时时间  sendbuf:发送 buffer  sendbuf\_size: 发送 buffer 长度  readbuf: 接收 buffer  readbuf\_size: 接收 buffer 长度 |
| 返回值 | 无 |
| 注意事项 | 无。 |

## NetworkConnect

|  |  |
| --- | --- |
| 格式 | int NetworkConnect(Network\* n, char\* addr, int port) |
| 描述 | 网络连接 |
| 参数说明 | client：客户端实例 |
| 返回值 | n：网络连接参数  addr：服务器 URL  port：服务器段口号 |
| 注意事项 | 无。 |

## MQTTConnect

|  |  |
| --- | --- |
| 格式 | Int MQTTConnect(MQTTClient\* c, MQTTPacket\_connectData\* options) |
| 描述 | MQTT 连接 |
| 参数说明 | c：MQTT 实例  options：连接参数，包括客户端标识符，用户名、密码、保连时间等 |
| 返回值 | 成功： 0 |
| 注意事项 | 连接之前需要给实例c中的结构体成员defaultMessageHandler（默认消息处理函数）赋值。 |

## MQTTPublish

|  |  |
| --- | --- |
| 格式 | int nwy\_MQTTPublish(MQTTClient\* c, const char\* topicName, MQTTMessage\* message) |
| 描述 | MQTT发布消息 |
| 参数说明 | topicName：topic 名称  message：消息体，包括：消息、消息长度、保留标志位等。 |
| 返回值 | 成功：0  失败：其他 |
| 注意事项 | 无。 |

## MQTTSubscribe

|  |  |
| --- | --- |
| 格式 | int MQTTSubscribe(MQTTClient\* c, const char\* topicFilter, enum QoS qos, messageHandler messageHandler) |
| 描述 | MQTT 订阅 topic，注册消息上报处理函数。 |
| 参数说明 | topicFilter：topic 名称  qos：服务质量等级  messageHandler：消息处理函数 |
| 返回值 | 成功：0  失败：其他 |
| 注意事项 | 无。 |

## MQTTSubscribe

|  |  |
| --- | --- |
| 格式 | int MQTTSubscribe(MQTTClient\* c, const char\* topicFilter, enum QoS qos, messageHandler messageHandler) |
| 描述 | MQTT 订阅 topic，注册消息上报处理函数。 |
| 参数说明 | topicFilter：topic 名称  qos：服务质量等级  messageHandler：消息处理函数 |
| 返回值 | 成功：0  失败：其他 |

## MQTTDisconnect

|  |  |
| --- | --- |
| 格式 | int MQTTDisconnect(MQTTClient\* c) |
| 描述 | MQTT 断开连接并释放资源 |
| 参数说明 | c：MQTT 实例 |
| 返回值 | 成功：0  失败：其他 |
| 注意事项 | 无。 |

## NetworkDisconnect

|  |  |
| --- | --- |
| 格式 | void NetworkDisconnect(Network\* n) |
| 描述 | 断开网络连接 |
| 参数说明 | N：网络参数 |
| 返回值 | 无 |
| 注意事项 | 无。 |

## MQTTYield

|  |  |
| --- | --- |
| 格式 | int MQTTYield(MQTTClient\* c, int timeout\_ms) |
| 描述 | 接收网络报文并将消息分发到用户的回调函数中。 |
| 参数说明 | c：MQTT 实例  timeout\_ms：接收报文超时时间 |
| 返回值 | 成功：0  失败：其他 |
| 注意事项 | 无。 |