Computer Science and Application 计算机科学与应用, 2023, 13(12), 2295-2304
Published Online December 2023 in Hans. https://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2023.1312229

Hans汉斯

基于MODBUS与MQTT协议自动转换的嵌入式智能系统设计

雷　妮[1]，胡鑫明[2,3]

[1]中国人民武装警察部队工程大学信息工程学院，陕西　西安
[2]西安西电数字科技有限公司，陕西　西安
[3]西安西电电力系统有限公司，陕西　西安

收稿日期：2023年11月20日；录用日期：2023年12月18日；发布日期：2023年12月25日

摘　要

为实现标准电力通信规约MODBUS到工业物联网标准规约MQTT的灵活转换，解决工业电力电子设备与物联网云平台通信互联问题，对MODBUS和MQTT协议的通信过程进行研究，设计了一种基于MODBUS与MQTT协议转换自动化的嵌入式智能系统。该系统采用ARM Cortex-A9嵌入式架构的Freescale i.MX6主控芯片开发应用程序，通过CP210x UART驱动模块和WIFI无线模块进行通信硬件层连接，支持远距离数据传输。本方案选择以C函数加动态库的软件接口编程方式分别实现MODBUS协议模块和MQTT协议模块，并设计采用双线程及数据格式转换实现了异步协议数据的透明转换，提高了系统运行效率。经过仿真试验验证，系统稳定可靠，设计满足设计技术规范要求。

关键词

MODBUS，MQTT，i.MX6，协议自动转换，接口编程，嵌入式智能系统

Embedded Intelligent System Design Based on Automatic Conversion of MODBUS and MQTT Protocol

Ni Lei[1], Xinming Hu[2,3]

[1]College of Information Engineering, Engineering University of PAP, Xi'an Shaanxi
[2]Xi'an XD Digital Technology Co., Ltd., Xi'an Shaanxi
[3]Xi'an XD Power System Co., Ltd., Xi'an Shaanxi

Received: Nov. 20th, 2023; accepted: Dec. 18th, 2023; published: Dec. 25th, 2023

文章引用: 雷妮, 胡鑫明. 基于 MODBUS 与 MQTT 协议自动转换的嵌入式智能系统设计[J]. 计算机科学与应用, 2023, 13(12): 2295-2304. DOI: 10.12677/csa.2023.1312229

Abstract

In order to realize the flexible conversion from MODBUS of standard power communication protocol to MQTT of industrial Internet of Things, and solve the problem of communication between industrial power electronic equipment and Internet of things cloud platform, this paper studies the communication process of MODBUS and MQTT protocol, and designs an embedded intelligent system based on the automation of MODBUS and MQTT protocol conversion. The system uses the Freescale i.MX6 master chip of the ARM Cortex-A9 embedded architecture to develop applications, and the MODBUS and MQTT protocol communication hardware layer connections are realized using the CP210x UART drive module and WIFI wireless module to support long-distance data transmission. Then the interface programming method of C function plus dynamic library is selected to realize Modbus protocol and MQTT protocol. The transparent conversion of asynchronous protocol data was realized by adopting the dual thread and data format conversion, which improves the operation efficiency of the system. The design has been verified by the simulation, the system is stable and reliable, and the simulation result meets the requirements of design specifications.

Keywords

MODBUS, MQTT, i.MX6, Automatic Protocol Conversion, Interface Programming Method, Embedded Intelligent System

1. 引言

文献[1]规范了 MQTT 协议，MQTT (Message Queuing Telemetry Transport，消息队列遥测传输)是基于发布/订阅模式的轻量级即时通信协议，已成为 OASIS (Organization for the Advancement of Structured Information Standards 结构化信息标准促进组织)标准规范。MQTT 协议使用 TCP/IP 提供网络连接，传输的消息分为主题(Topic)和负载(Payload)两部分，通过主题对消息进行分类，并允许使用通配符订阅主题；支持三种消息发布 QoS (Quality of Service，服务质量)；使用遗言机制 Will 通知客户端异常中断；协议采用交换控制帧来工作，一个控制帧由固定头(Fixed header)、可变头(Variable header)和消息体(payload)三部分构成，固定长度的头部是 2 字节，开销很小，非常适合需要低功耗和网络带宽有限的物联网场景。本研究实现了标准帧格式发布订阅功能。

文献[2]规范了串行链路和 TCP/IP 两个通信规程中使用 Modbus 应用层协议和服务规范，Modbus 是一个请求/应答的应用层报文传输协议，提供 PDU (Protocol Data Unit 协议数据单元)的功能码规定的服务，用于在通过不同类型的总线或网络连接的设备之间的客户机/服务器通信。Modbus 协议的通讯格式主要分 ASCII、RTU、TCP 等，其中 RTU 格式由于传输效率高被大多数厂商采用[3]。此协议支持传统的 RS-232、RS-422、RS-485 和光纤、无线设备，本研究采用 RS-232 实现 Modbus-RTU 异步串行通信，系统采用主/从技术，通过唯一设备地址完成通讯任务，主设备发出请求从设备根据主设备发出的请求消息作出响应。

在工程应用中，经常需要将符合通用电力规约的设备传来的 MODBUS 协议数据提取出来，经由物联网 MQTT 协议转换传输到服务器做后续处理，实现设备数据到云平台的传递。该智能系统平台基于此工程应用通过嵌入式方式实现了标准电力通信规约 MODBUS 到工业物联网标准规约 MQTT 的灵活转换，解决工业电力电子设备无法直接与物联网云平台通信问题。

2. 设计方案选择

协议算法常见的实现方案有两种：

(1) 依赖于 MQTT 及 MODBUS 协议具体实现，针对实现编程。研究各协议机制，从底层代码编写，实现各协议功能。这种方式灵活度高，可实现自定义的数据帧收发功能，但开发周期长，对开发人员编程能力要求较高，实现协议完全标准化可靠性不高，开发调试过程复杂。

(2) 依赖于开源库 API 库函数，针对接口编程。在开源库基础上进行协议开发，加快开发进程。根据需求通过设置不同参数调用 API 库函数接口，设计协议传输数据内容，对不同的通讯格式保持一致的代码形式，将开发人员大量精力从编写底层代码的工作转移到上层功能应用开发。基于 ARM 嵌入式平台实现的 MQTT 协议开源库和 MODBUS 协议开源库可参考的资料较多，借助于 ARM 嵌入式测试平台即可完成所有软件的开发，对软件平台要求和开发难度相对较低，通信软件的扩展灵活性高。

3. 设计实现

综合评估开发的难度和现有的开发软件平台，该智能终端系统采用上文中提到的第二种方案，针对接口编程，以 C 函数加动态库的形式进行 ARM 嵌入式工控主板协议转换应用程序开发。本方案选择 libmodbus 库和 libmosquitto 库作为协议开发基础，见图 1：嵌入式智能系统协议转换设计实现流程图。在 PC 机上进行协议功能配置并交叉编译，将编译成功的 libmosquitto.so、libmodbus.so 及所依赖动态库移植 ARM 嵌入式智能系统进行 LD_LIBRARY_PATH 库路径配置；将交叉编译可执行程序 Mosquitto 安装在 ARM 嵌入式智能系统平台，进行 PATH 环境变量路径配置；调用 API 函数库接口，实现协议转换功能应用程序开发并进行交叉编译，移植嵌入式智能系统进行功能调试。

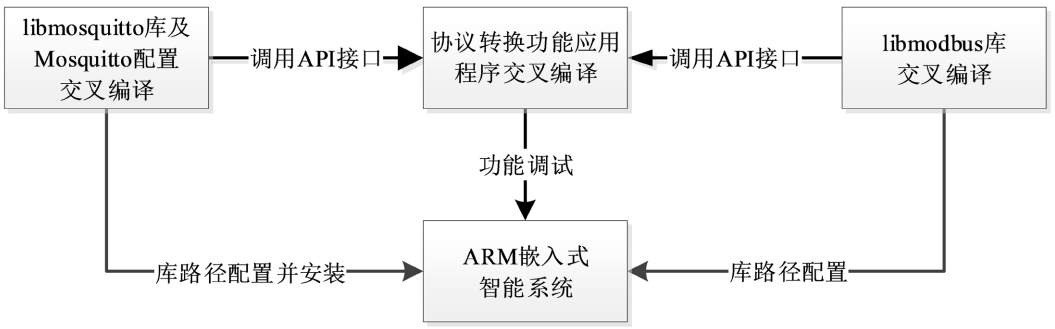


Figure 1. Schematic diagram of embedded intelligent system implementation
图 1. 嵌入式智能系统实现示意图

该系统选用嵌入式软硬件平台为基于 ARMCortex-A9 架构的 Freescale i.MX6 Quad 主控芯片，在 windows 系统安装虚拟机搭建嵌入式 Linux 开发环境。

文献[4]介绍了四核 Freescale i.MX6 Quad 系列平台特点。每个内核运行频率高达 1.2 GHz，带有 1 MB L2 缓存，和 64 位 DDR3 或 2 通道、32 位 LPDDR2 控制器，以更低的功耗提供更高的性能，适用于对功率有严格要求的应用。这个系列的器件集成了 FlexCAN、UART、Ethernet、USB 和 SATA-2 等，具有卓

越的连接性，其广泛的集成 I/O 提供了高集成度和可扩展性，易于 4G 模块功能扩展及串口驱动安装，可降低设计复杂度。

3.1. MQTT 协议实现

Mosquitto [5]是一个开源的轻量级的 C 实现消息代理服务器，完全兼容了 MQTT 3.1 和 MQTT 3.1.1 协议。libmosquitto 作为 Mosquitto 开源代码的一部分，主要用来实现 MQTT 协议栈和数据包通讯功能。文献[6]介绍了 Mosquitto 的消息推送机制，并针对大文件类型消息对消息机制进行改进。根据本研究实际应用，传输的消息长度不会高于 25 MB，因此直接保留原有即时性高的传输方式。

3.1.1. Mosquitto 编译及配置

本研究使用 Mosquitto 源码，分别通过 config.mk 和 mosquitto.conf 进行安装选项和启动参数配置。Mosquitto 编译动态库 libmosquitto.so 依赖 libuuid.so、libcrypto.so 及 libssl.so 动态库，同时生成 mosquitto、mosquitto_passwd、mosquitto_pub 和 mosquitto_sub 四个工具，分别用于启动代理、管理密码、发布消息和订阅消息。

3.1.2. 订阅/发布设计

一个完整的 Mosquitto 通信包括一个代理服务器，一个发布者 Publisher 和一个订阅者 Subscriber。其工作过程分为以下四个步骤，基本模型如图 2。

(1) 启动服务 mosquitto。

(2) 订阅者 Subscriber 通过 mosquitto_sub 订阅指定 Topic 的消息。

(3) 发布者 Publisher 通过 mosquitto_pub 发布指定 Topic 的消息。

(4) 代理服务器把 Topic 消息推送到订阅者。

Figure 2. Mosquitto basic model diagram
图 2. Mosquitto 基本模型图

本研究设计 MQTT_t 结构体用来存放 sub_topic 订阅主题、发布主题 pub_topic、发送缓冲区 mqtt_buffer、接收缓冲区 mqtt_read_buffer 及 will_topic 遗嘱主题等参数，主要分为初始化结构体、订阅数据及发布数据三部分实现。

```
typedefstruct {
charsub_topic[MQTT_TOPIC_SIZE];
charpub_topic[MQTT_TOPIC_SIZE];
charmqtt_buffer[MQTT_BUF_SIZE];
charmqtt_read_buffer[MQTT_BUF_SIZE];
charwill_topic[MQTT_TOPIC_SIZE];
}MQTT_t;
```

(1) 结构体的初始化

init 部分实现代码，初始化订阅及发布主题信息：

```
voidervry_mqtt_init(MQTT_t *mqtt)
```

```
{
sprintf(mqtt->sub_topic,"/%s/to_condev",company);
sprintf(mqtt->pub_topic,"/%s/to_webapp",company);
}
```

(2) 订阅数据

mosquitto_sub 部分实现代码，订阅指令编码：

```
voidervry_mosquitto_sub(MQTT_t * mqtt)
{
strcat(strCmd, "mosquitto_sub -h ");
strcat(strCmd, MQTT_HOST);
strcat(strCmd, " -t ");
strcat(strCmd, mqtt->sub_topic);
fp = popen(strCmd, "r") ;
memset(mqtt->mqtt_read_buffer,0,MQTT_BUF_SIZE);
fgets(mqtt->mqtt_read_buffer,MQTT_BUF_SIZE,fp);
}
```

(3) 发布数据

mosquitto_pub 部分实现代码，发布指令编码：

```
voidervry_mosquitto_pub(MQTT_t *mqtt,void *pbuf)
{
strcat(strCmd, "mosquitto_pub -h ");
strcat(strCmd, MQTT_HOST);
strcat(strCmd, " -t ");
strcat(strCmd, mqtt->pub_topic);
strcat(strCmd, " -m ");
strcat(strCmd, pbuf);
system(strCmd);
}
```

3.2. Modbus 通信功能实现

Modbus 的核心是一个串行通信协议，采用主从模式，主机向从机发送请求，从机予以回复[3]。文献[7]采用 VC 编写 Modbus RTU 通讯，重点说明了数据的传输过程和 CRC 冗余循环码校验功能。文献[8]采用 C++开发 Modbus RTU 及 TCP/IP 通信协议，实现嵌入式系统数据采集及上送服务器功能。本研究使用 libmodbus 库进行 Modbus-RTU 开发。libmodbus 是一个快速且可移植的 Modbus 库，支持传统的 RS-232、RS-422、RS-485 和以太网设备。Modbus-RTU 主程序设计流程如图 3。

(1) libmobus 实例创建，并设置参数，调用库函数 modbus_new_rtu()实现功能；

(2) 设置从站 ID，调用库函数 modbus_set_slave()实现功能；

(3) 建立连接，调用库函数 modbus_connect()实现功能；

(4) Modbus 读取数据功能码与库函数对应关系如表 1 所示；

Table 1. Correspondence table of read data function code and library function
表 1. 读取数据功能码与库函数对应关系表

功能码	地址	库函数名称
01	00001-09999	modbus_read_bits
02	10001-19999	modbus_read_input_bits
03	40001-49999	modbus_read_registers
04	30001-39999	modbus_read_input_registers

(5) Modbus 写数据功能码与库函数对应关系如表 2 所示；

Table 2. Correspondence table of write data function code and library function
表 2. 写数据功能码与库函数对应关系表

功能码	地址	库函数名称
05	00001-09999	modbus_write_bit
06	40001-49999	modbus_write_register
15	00001-09999	modbus_write_bits
16	40001-49999	modbus_write_registers

(6) 关闭连接，调用库函数 modbus_close()及 modbus_free()释放 modbus 资源；

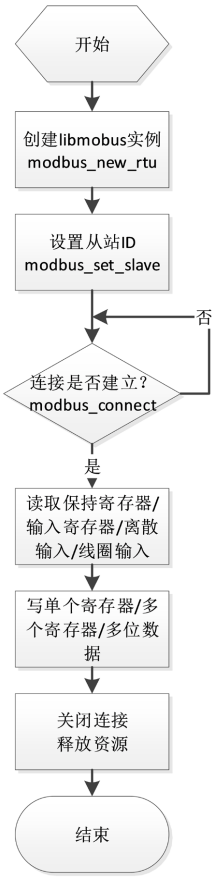


Figure 3. Flow chart of Modbus-RTU main program design
图 3. Modbus-RTU 主程序设计流程图

3.3. 协议间数据收发自动转换设计

本研究设计双线程实现协议间数据收发自动转换功能，更高效的利用 CPU，加快应用程序的响应，系统软件设计流程如图 4 所示。线程 thread_read_pub 实现 Modbus-RTU 采集底层数据格式转换后经 MQTT 上送服务器功能；线程 thread_sub_write 实现 MQTT 订阅主题数据格式转换后经 Modbus-RTU 发送至底层控制装置。

各个线程间有方便的通信和数据交换机制，可共享数据空间，一个线程的数据可以直接为另一个线程使用，快捷而且方便。本研究定义 MODBUS_t 结构体来存放 Modbus 接收发送缓存数据。

```
typedefstruct {
uint16_t modbus_read_buffer[BUF_SIZE];
uint16_t modbus_write_buffer[BUF_SIZE];
}MODBUS_t;
```

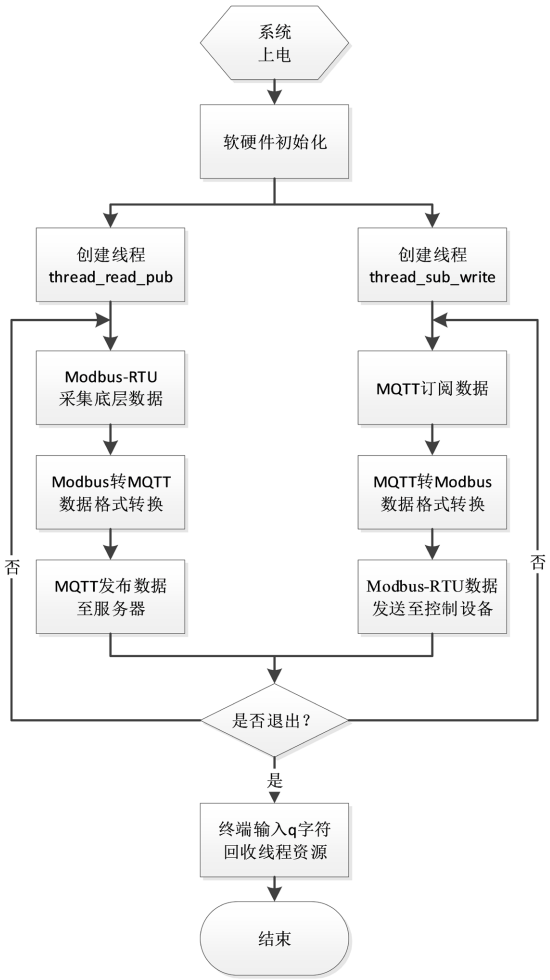


Figure 4. System software design flow chart
图 4. 系统软件设计流程图

本研究要解决协议自动转换功能，核心问题是解决协议数据格式转换。文献[9]采用 JSON 模块解析判断 Modbus 协议方式(RTU/TCP)来连通 Modbus 和 MQTT 模块。本研究应用需求采用 RTU 协议方式，

根据 3.1.2 设计的 MQTT_t 结构体及 Mosquitto 数据格式要求，针对 Modbus 转 MQTT 使用格式化输出 sprintf()函数将 Modbus 无符号整型数据转为 Mosquitto 指令可辨识的字符串类型，实现字符转字符串功能；针对 MQTT 转 Modbus 使用标准 atoi()函数将 Mosquitto 订阅指令接收到的字符串类型转换为 Modbus 无符号整型数据类型，实现字符串转字符功能，最终实现一帧 Modbus 数据与多个 MQTT 主题发布订阅灵活转换功能，简便可靠且易于操作。

4. 功能验证

该嵌入式智能系统采用 OKMX6X-C 开发平台，基于 Freescale i.MX6 Quad 处理器，ARMCortex-A9 架构，每个内核运行频率高达 1 GHz，支持串口、USB、通用以太网口及 WIFI 模块。

4.1. 试验环境搭建

系统功能试验环境如图 5，嵌入式智能系统平台通过内核配置安装 CP210x UART 驱动模块采用 USB 转 RS232 方式，与具有 Modbus-RTU 从站功能的工业电力电子设备装置通信，实现 Modbus 串口通信数据采集及发送；与物联网云平台采用 WIFI 模块通信，根据物联网云平台发布订阅数据，通过 TCP/IP 实现 MQTT 把数据上传到服务器或接收服务器数据，完成 Modbus 与 MQTT 不同协议数据灵活转换功能。

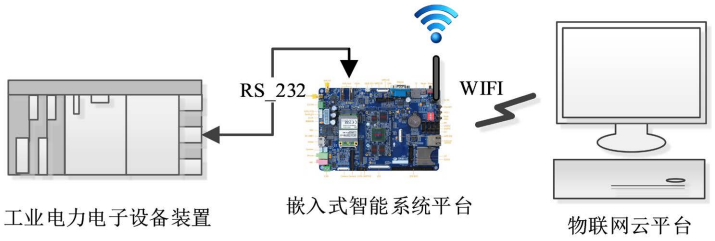


Figure 5. Establishment of system function test environment
图 5. 系统功能试验环境搭建

该系统使用/MQTT/to_webapp0 至/MQTT/to_webapp9 共十个主题来订阅数据，使用/MQTT/to_condev0 至/MQTT/to_condev9 共十个发布主题发布数据，使用服务器地址为 192.168.2.140。具有 Modbus-RTU 从站功能的工业电力电子控制设备接收该智能系统主站的命令包、回送数据包，实现 Modbus 通讯协议通信，支持功能 01，02，03，04，05，06，15，16，22 和 23，监视串口数据。

4.2. MQTT 转 MODBUS 协议功能

(1) 该系统订阅/MQTT/to_condev 10 个主题，服务器向该系统发送消息，发布数据十进制表示分别为 11、22、33、…99、100，如图 6 所示。

历史记录 最新主题消息

事件	主题	消息	服务质量	已保留
已连接				
已发布	/MQTT/to_condev0	11	0	否
已发布	/MQTT/to_condev1	22	0	否
已发布	/MQTT/to_condev2	33	0	否
已发布	/MQTT/to_condev3	44	0	否
已发布	/MQTT/to_condev4	55	0	否
已发布	/MQTT/to_condev5	66	0	否
已发布	/MQTT/to_condev6	77	0	否
已发布	/MQTT/to_condev7	88	0	否
已发布	/MQTT/to_condev8	99	0	否
已发布	/MQTT/to_condev9	100	0	否

Figure 6. Server publishes MQTT data
图 6. 服务器发布 MQTT 数据

(2) 该系统数据协议自动转换程序运行，接收服务器 192.168.2.140 发布的数据，进行数据格式转换，将数据进行打包整合通过/dev/ttyUSB0 串口设备经一帧 Modbus-RTU 协议报文发送出去，完成 MQTT 转 Modbus-RTU 数据格式转换功能。

(3) 具有 Modbus-RTU 从站功能的工业电力电子设备装置收到该系统主站发送的 RX 报文，发送回复报文 TX 进行确认，RX 报文与 MQTT 发布的 10 个协议点数据一一对应，如图 7 所示，000Bh 对应十进制 11，0016h 对应十进制 12，以此类推，可知 Modbus 从站控制设备收到服务器发送过来的 MQTT 协议数据，由此可验证 MQTT 转 Modbus-RTU 数据格式转换功能正确。

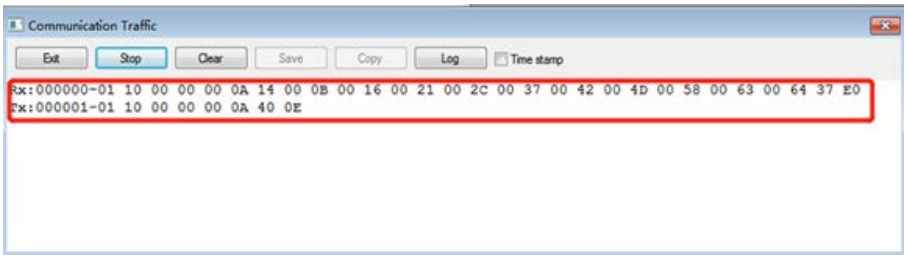


Figure 7. Modbus slave station control equipment receives Modbus-RTU messages
图 7. Modbus 从站控制设备接收 Modbus-RTU 报文

4.3. MODBUS 转 MQTT 协议功能

(1) 该系统采集底层数据，向具有 Modbus-RTU 从站功能的工业电力电子设备装置发起 Modbus 读 03 功能码报文，从站控制设备发送回复报文 TX 进行响应，其发送分别为 0064h、0063h、0058h、…、000Bh 等 10 个协议点数据，如图 8 所示。

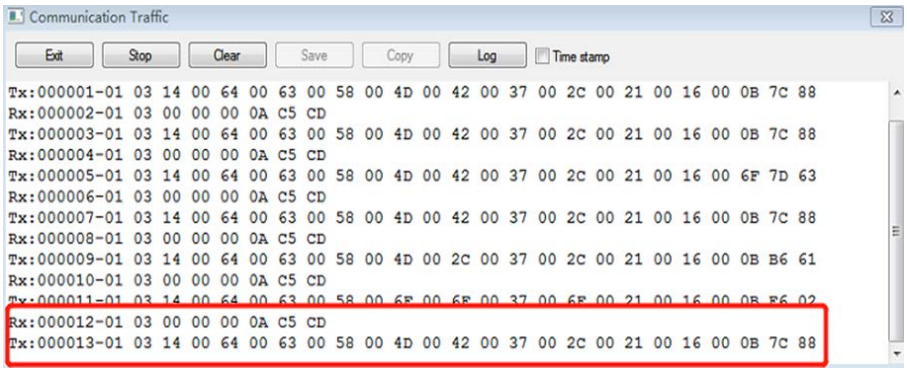
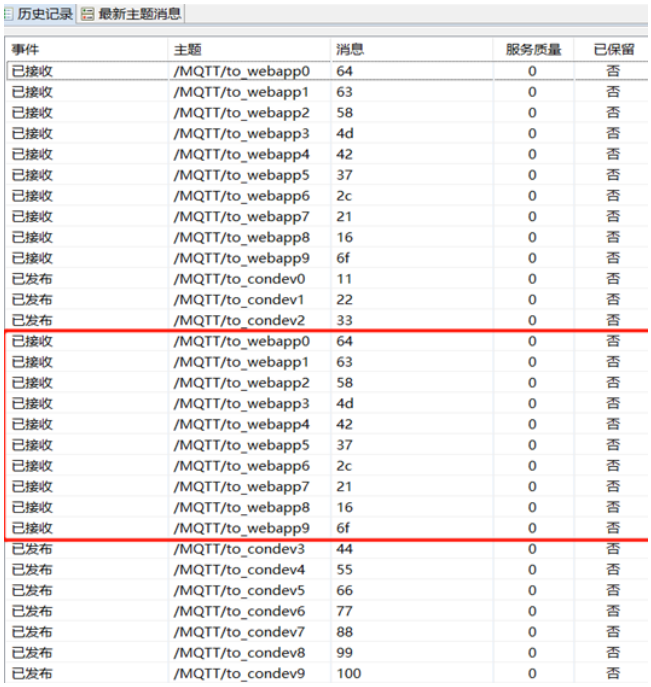


Figure 8. Modbus slave responds to the master station and sends Modbus-RTU message interface
图 8. 从站响应主站并发送 Modbus-RTU 报文

(2) 该系统主站收到 Modbus 回复数据报文，通过数据格式转换，将一帧 Modbus-RTU 数据转换为/MQTT/to_webapp 10 个发布主题发送至服务器，实现 Modbus-RTU 转 MQTT 数据格式转换功能。

(3) 该系统向服务器发送消息，发布/MQTT/to_webapp10 个主题，服务器订阅数据如图 9 所示，与 Modbus 从站设备装置发送的报文数据内容一一对应，分别为 0064h、0063h、0058h、…、000Bh，可知服务器已接收到 Modbus 从站控制设备发送的数据，由此可验证 Modbus-RTU 转 MQTT 功能块的正确性。



历史记录 | 最新主题消息

事件	主题	消息	服务质量	已保留
已接收	/MQTT/to_webapp0	64	0	否
已接收	/MQTT/to_webapp1	63	0	否
已接收	/MQTT/to_webapp2	58	0	否
已接收	/MQTT/to_webapp3	4d	0	否
已接收	/MQTT/to_webapp4	42	0	否
已接收	/MQTT/to_webapp5	37	0	否
已接收	/MQTT/to_webapp6	2c	0	否
已接收	/MQTT/to_webapp7	21	0	否
已接收	/MQTT/to_webapp8	16	0	否
已接收	/MQTT/to_webapp9	6f	0	否
已发布	/MQTT/to_condev0	11	0	否
已发布	/MQTT/to_condev1	22	0	否
已发布	/MQTT/to_condev2	33	0	否
已接收	/MQTT/to_webapp0	64	0	否
已接收	/MQTT/to_webapp1	63	0	否
已接收	/MQTT/to_webapp2	58	0	否
已接收	/MQTT/to_webapp3	4d	0	否
已接收	/MQTT/to_webapp4	42	0	否
已接收	/MQTT/to_webapp5	37	0	否
已接收	/MQTT/to_webapp6	2c	0	否
已接收	/MQTT/to_webapp7	21	0	否
已接收	/MQTT/to_webapp8	16	0	否
已接收	/MQTT/to_webapp9	6f	0	否
已发布	/MQTT/to_condev3	44	0	否
已发布	/MQTT/to_condev4	55	0	否
已发布	/MQTT/to_condev5	66	0	否
已发布	/MQTT/to_condev6	77	0	否
已发布	/MQTT/to_condev7	88	0	否
已发布	/MQTT/to_condev8	99	0	否
已发布	/MQTT/to_condev9	100	0	否

Figure 9. Server subscription receives MQTT data
图 9. 服务器订阅接收 MQTT 数据

5. 结束语

该系统采用高性能工业 Freescale i.MX6 作为核心控制单元，基于 ARM 嵌入式 Linux 实时操作系统运行环境，采用 C 函数加动态库的接口编程形式实现了 MODBUS-RTU 通信协议及 MQTT 通信协议，并采用双线程及数据格式转换实现了异步协议的相互自动转换功能设计，解决传统工业电力电子设备无法与物联网云平台直接通信互联问题，经试验验证，系统运行稳定，能够满足数据传输的实时性、可靠性要求，同时系统便于升级和标准化，可以满足不同的工程应用需求。

参考文献

[1] Banks, A. and Gupta, R. (2014) OASIS Standard MQTT Version3.1.1. http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html

[2] 丛航, 孙昕, 欧阳劲松. 基于 Modbus 协议的工业自动化网络规范——基于串行链路和 TCP/IP 的 Modbus 应用协议[J]. 仪器仪表标准化与计量, 2003(1): 14-17+48.

[3] 樊星男. 基于 Modbus/TCP 协议的 Modicon M238 通信网络设计[J]. 机电技术, 2016(2): 57-60.

[4] (2012) NXP. i.MX 6 Series of Applications Processors. NXP, Nederland. www.nxp.com/iMX6Series

[5] (2020) Mosquitto. https://mosquitto.org/

[6] 曾昂, 李宁, 严俊. Mosquitto 大文件传输方式的研究与改进[J]. 2017, 53(4): 123-127.

[7] 李喜东, 刘波涛, 刘刚. Modbus RTU 串行通讯协议在工业现场的应用[J]. 2005, 24(7): 37-40.

[8] 邓婷, 范润宇. 远程电力抄表系统的网关通信设计与实现[J]. 2020, 23(3): 60-62, 52.

[9] 胡存, 骆德汉, 童怀. 基于 Modbus 与 MQTT 融合工业能耗网关系统设计[J]. 物联网技术, 2019(4): 49-54.