Document Title: CM_FW modbus application module design description of Safety Control System

Document Number: 15-Q04-000134

Project Number: CT-RD-1601

Project Name: First phase of Safety Control System Development Project

Material Number: N/A

Document Version: A

Classification Level: Highly secret

Document Status: CFC

Controlled Status: Under control

Prepared by: Wang Dong 2016-11-01

Checked by: Zhu Genghua 2016-11-30

Countersigned by: Li Qi, Liu Yang

Approved by: Wen Yiming 2016-12-30

# Revision History

| No. | Relevant Chapter | Change Description | Date | Version Before Change | Version After Change | Prepared by | Checked by | Approved by |
|---|---|---|---|---|---|---|---|---|
| 1 | | Document created | 2016-11-1 | None | A | Wang Dong | Zhu Genghua | Wen Yiming |
| 2 | | | | | | | | |
| 3 | | | | | | | | |
| 4 | | | | | | | | |
| 5 | | | | | | | | |

**Relationship between this version and old versions: None.**

文件名称：安全控制系统 CM_FW Modbus 应用模块设计说明书

文件编号：15-Q04-000134

项目编号：SF-RD-1601

项目名称：安全控制系统开发项目一期

物料编号：

版本号/修改码：A

文件密级：机密

文件状态：CFC

受控标识：受控

拟制：王 东 2016 年 11 月 1 日

审核：朱耿华 2016 年 11 月 30 日

会签：李 琦 刘 阳

批准：温宜明 2016 年 12 月 30 日

# 修订页

| 编号 | 章节名称 | 修订内容简述 | 修订日期 | 订前版本 | 订后版本 | 拟制 | 审核 | 批准 |
|---|---|---|---|---|---|---|---|---|
| 1 | | 创建 | 2016-11-1 | | A | 王 东 | 朱耿华 | 温宜明 |
| 2 | | | | | | | | |
| 3 | | | | | | | | |
| 4 | | | | | | | | |
| 5 | | | | | | | | |
| 6 | | | | | | | | |
| 7 | | | | | | | | |
| 8 | | | | | | | | |
| 9 | | | | | | | | |
| 10 | | | | | | | | |
| 11 | | | | | | | | |
| 12 | | | | | | | | |
| 13 | | | | | | | | |
| 14 | | | | | | | | |
| 15 | | | | | | | | |
| 16 | | | | | | | | |

**本版本与旧文件（版本）的关系：**

# Content 目录

# 1 Document overview 文档概述

## 1.1 Introduction 综述

This document describes the design description of modbus function of CM_FW of Safety Control System. The document describes the overall concept of the function of the module, and then the sub-function of the modules are described in detail.

This document is the output of module design phase of CM_FW, and is the input for the follow-up coding phase.

本文档描述了安全控制系统中 CM_FW Modbus 模块的设计方案。文档首先描述了模块功能的总体设计思路，然后将模块功能划分为若干子功能并进行详细说明。

本文档是 CM_FW 模块设计的输出，也是后续编码的输入。

## 1.2 Reference 参考文档

### 1.2.1 Project documents 内部参考文档

[1] Embedded software safety concept of Safety Control System [505], 15-Q02-000059

[1] 安全控制系统嵌入式软件安全概念说明书 [505], 15-Q02-000059

[2] PM_FW software overall design description of safety control system [506], 15-Q02-000074

[2] 安全控制系统 PM_FW 总体设计说明书 [506], 15-Q02-000074

## 1.3 Terms and abbreviations 术语和缩略语

### 1.3.1 Terms 术语

Table 1-1 Terms

表 1-1 术语

| No.<br>序号 | Term<br>术语 | Description<br>解释 |
|---|---|---|
| 1. | IP_BUS | Communication between PM and IO modules.<br>PM 与 IO 模块之间的通讯总线。 |
| 2. | CM_BUS | Communication between PM and CM.<br>PM 与 CM 之间的通讯总线。 |
| 3. | PM_BUS | Communication between PMs.<br>PM 之间的通讯总线。 |
| 4. | System Net | Communication between control station and PC.<br>控制站与上位机之间的通讯网络。 |
| 5. | Safety Net | Safe communication between control stations. |

| | | |
|---|---|---|
| | | 控制站之间的安全通讯。 |
| 6. | Control station<br>控制站 | A set of triple redundant control system, which includes triple redundant PMs and IO modules under control.<br>一套三冗余的控制系统，包含三冗余 PM 和 PM 控制的各种 IO 模块。 |
| 7. | System response time<br>系统响应时间 | Time interval from the moment that transition of demand signal generated at input ETP to the moment that transition of response signal generated at output ETP.<br>从系统输入端子板上产生需求信号跳变的时刻到输出端子板上产生相应的响应信号跳变之间的时间。 |
| 8. | Control cycle<br>控制周期 | Time interval between adjacent two runs of user program execution.<br>PM 两次执行用户程序间隔时间。 |
| 9. | Project<br>工程 | Files which contain configuration information for control station and generated by IEC 61131 configuration software. These files contain all the information required by control station to implement control, including user control program (binaries) to be loaded and executed as well as configuration information of task, CM, PM and IO modules.<br>IEC 61131 组态软件在完成编译后，为控制站生成的组态信息文件，该文件包含可加载执行的用户控制程序（二进制程序）、任务配置信息、CM 配置信息、PM 配置信息和 IO 模块配置信息等各种控制站完成控制所需的信息。 |
| 10. | Source project<br>源工程文件 | Source file of the project before compiling.<br>工程在编译前的源文件。 |
| 11. | User program<br>用户程序 | Part of project which contain user control program (binaries) to be loaded and executed and configuration information of task.<br>工程中的一部分：可加载执行的用户控制程序（二进制程序）和任务配置信息。 |

### 1.3.2 Abbreviations 缩略语

Table 1-2 Abbreviations

表 1-2 缩略语

| No.<br>序号 | Abbreviation<br>缩略语 | English description<br>英文 | Chinese description<br>中文 |
|---|---|---|---|
| 1. | PM | Processor Module | 主处理器模块 |
| 2. | CM | Communication Module | 通讯模块 |
| 3. | BI | Bus Interface Module | 总线接口模块 |
| 4. | AI | Analog Input Module | 模拟量输入模块 |
| 5. | AO | Analog Output Module | 模拟量输出模块 |

| | | | |
|---|---|---|---|
| 6. | DI | Digital Input Module | 数字量输入模块 |
| 7. | DO | Digital Output Module | 数字量输出模块 |
| 8. | OSP | Over Speed Protect Module | 超速保护模块 |
| 9. | SOE | Sequence Of Events | SOE 事件 |
| 10. | SIL | Safety Integrity Level | 安全完整等级 |
| 11. | PW | Power Module | 电源模块 |
| 12. | OPC | OLE for Process Control | 用于过程控制的对象链接与嵌入式技术 |
| 13. | UP | User Program | 用户程序 |

# 2 Module overview 模块概述

The location of the modbus application module (marked red) in the software hierarchy is shown below.

Modbus 应用模块（标红）在软件层次中的位置如下图所示。

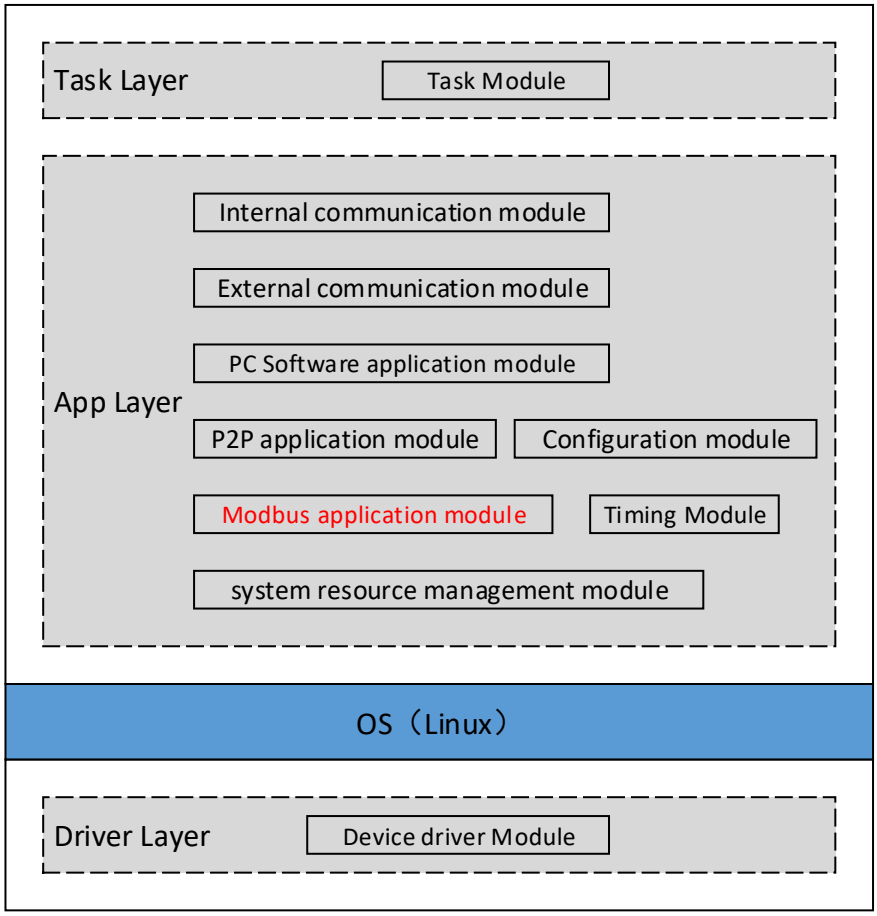


Figure 2-1 the location of the modbus module
图 2-1 模块位置

Modbus module is mainly used to process modbus communication.

Modbus 模块主要用于实现 Modbus 通讯功能。

# 3 Module design 模块设计

## 3.1 Function description 功能描述

The main functions are as follows:

主要功能如下：

- Manage configuration: Modbus master configuration, modbus slave configuration, and port (network/serial) configuration;

  管理配置信息：Modbus 主站配置，Modbus 从站配置和端口（网口/串口）配置；

- Make modbus request message, and send to the modbus slave station;

  构造 Modbus 请求帧，并发送给相应的 Modbus 从站；

- Decode response message from the modbus slave station, and send data to PM;

  解析来自 Modbus 从站的应答帧，将来自 Modbus 从站的数据转发到 PM；

- Decode request message from the modbus master station, and send data to PM;

  解析来自 Modbus 主站的请求帧，将来自 Modbus 主站的数据转发到 PM；

- Make modbus response message, and send to the modbus master station.

  构造 Modbus 应答帧，并应答给相应的 Modbus 主站。

## 3.2 Design concept 设计思路

### 3.2.1 Data flow 数据流

#### 3.2.1.1 Modbus TCP master station data flow Modbus TCP 主站数据流

Take the network port 1 as an example, data flow is shown below:

以网口 1 为例进行说明，数据流如下图所示：

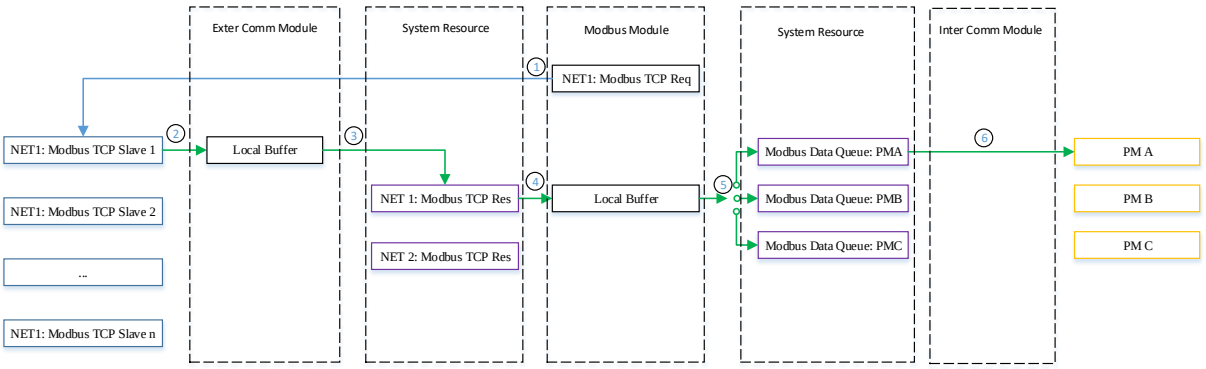


Figure 3-1 data flow of modbus TCP master station

图 3-1 Modbus TCP 主站数据流

#### 3.2.1.2 Modbus serial master station data flow Modbus 串口主站数据流

Take the serial port 1 as an example, data flow is shown below:

以串口 1 为例进行说明，数据流如下图所示：

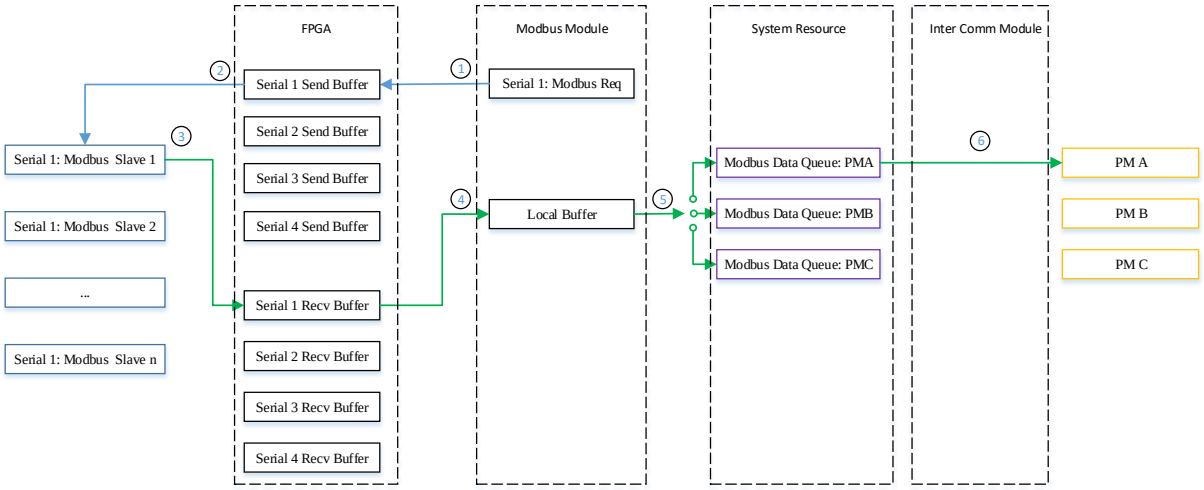


Figure 3-2 data flow of modbus serial master station

图 3-2 Modbus 串口主站数据流

#### 3.2.1.3 Modbus TCP slave station data flow Modbus TCP 从站数据流

Take the network port 1 as an example, data flow is shown below:

以网口 1 为例进行说明，数据流如下图所示：

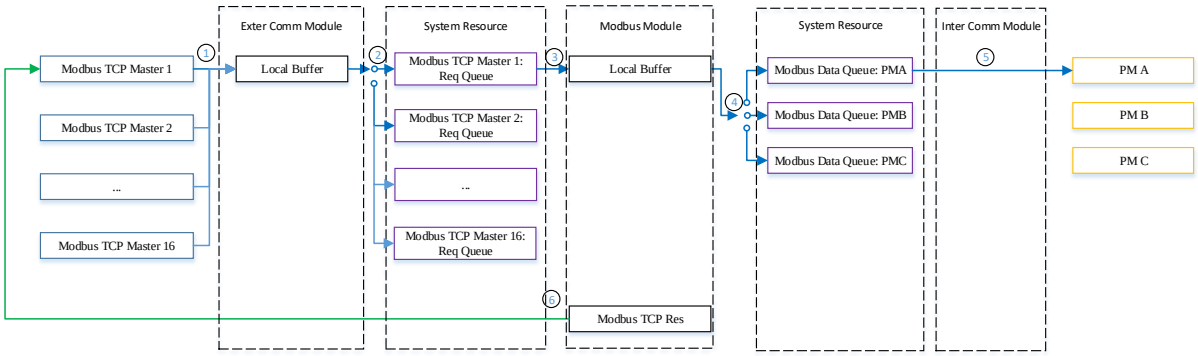


Figure 3-3 data flow of modbus TCP slave station

图 3-3 Modbus TCP 从站数据流

#### 3.2.1.4 Modbus serial slave station data flow Modbus 串口从站数据流

Take the serial port 1 as an example, data flow is shown below:

以串口 1 为例进行说明，数据流如下图所示：

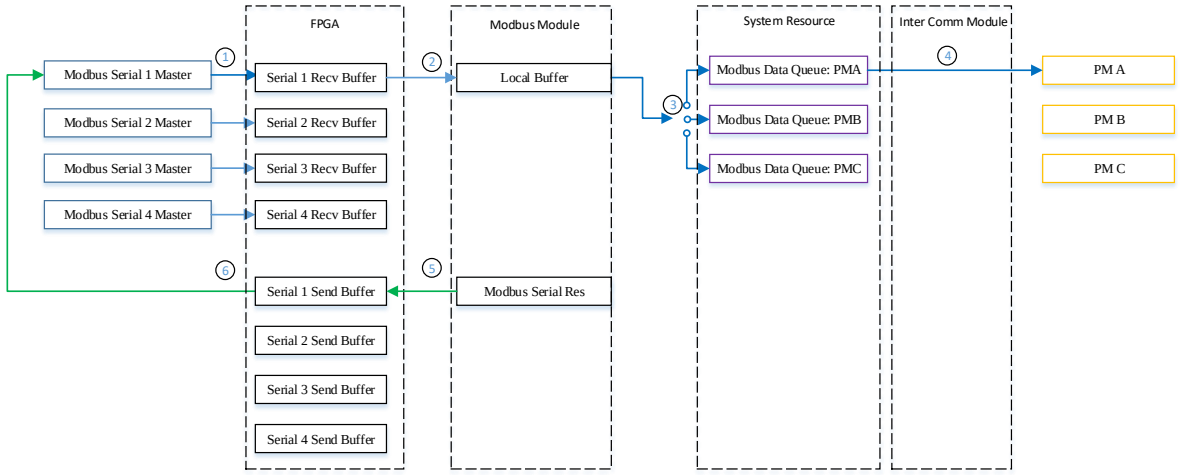


Figure 3-4 data flow of modbus serial slave station

图 3-4 Modbus 串口从站数据流

### 3.2.2 Configuration management 配置信息管理

For each configuration, module stores two versions locally and records the version in use. In each period, the module compares the version stored in the system resource with the local version currently in use: If they are different and real data version is changed too, copy it to local and switch to the new version.

对于每种配置，模块内部均存储两个版本，并记录正在使用的版本。模块每周期均检查系统资源中的相应配置的版本是否发生了变化，如果变化且实时数据的版本也发生了改变，则拷贝到本地并切换。

The module handles the changed parameters when switch.

发生切换时，对变化的参数进行处理。

### 3.2.3 Modbus master port Modbus主站端口

When the port is configured as a modbus master port, poll its slave station(s) according to its configuration information. Send one request message in each period, and then wait for response from the specific slave station.

端口配置为 Modbus 主站端口时，根据其配置信息，轮询相应的从站。每周期最多发送一个请求帧，然后等待相应从站的应答。

When receiving response message from the specific slave station, if it is read request, package read data and send to PM.

接收到相应从站的应答帧后，如果是读请求的应答，则打包读到的数据，并发送到 PM。

### 3.2.4 Modbus slave port Modbus从站端口

When the port is configured as a modbus slave port, decode all request messages of one master station in each period. If it is write request, package write data and send to PM.

端口配置为 Modbus 从站端口时，每周期最多处理一个 Modbus 主站的所有请求帧。如果是写请求，则打包收到的数据，并发送到 PM。

### 3.2.5 Format definition of PM modbus frame PM modbus数据帧格式定义

#### 3.2.5.1 Data frame of master port 主站端口数据帧

Table 3-1 Data frame of master port

表 3-1 主站端口数据帧

| Frame header<br>帧头 | Modbus data<br>Modbus 数据 |
|---|---|
| 28 bytes<br>28 字节 | n bytes. Note: Each BOOL variable is one bit.<br>n 字节。注：一个 BOOL 量占一位。 |

Table 3-2 Frame header

表 3-2 帧头

| Field<br>数据域 | Type<br>类型 | Length<br>长度 | Value<br>数值 | Description<br>描述 |
|---|---|---|---|---|
| Port type | uint8_t | 1 byte | 0 | Port type: 0-Modbus master; 1-Modbus slave<br>端口类型：0-Modbus 主站；1-Modbus 从站 |
| Reserve 1 | uint8_t | 1 byte | | Reserve 1<br>预留 1 |
| Data length | uint16_t | 2 bytes | n | Modbus data length (bytes)<br>Modbus 数据的长度（字节） |
| CM slot ID | uint8_t | 1 byte | | CM slot ID<br>CM 模块槽位号 |
| Port ID | uint8_t | 1 byte | | Port ID: 0-3, COM1-4; 4-5, Net1-2<br>端口编号：0-3，串口 1-4；4-5，网口 1-2 |
| Reserve 2 | uint8_t[] | 2 bytes | | Reserve 2<br>预留 2 |
| Block ID | uint32_t | 4 bytes | | Modbus master block ID<br>Modbus 主站块号 |
| Project ID | uint32_t | 4 bytes | | Project ID<br>工程 ID |
| Main version | uint32_t | 4 bytes | | Main version<br>主版本 |
| Minor version | uint32_t | 4 bytes | | Minor version<br>次版本 |
| Project CRC | uint32_t | 4 bytes | | Project CRC<br>工程 CRC |

#### 3.2.5.2 Data frame of slave port 从站端口数据帧

Table 3-3 Data frame of slave port

表 3-3 从站端口数据帧

| Frame header | Modbus data |
|---|---|

| 帧头 | Modbus 数据 |
|---|---|
| 28 bytes<br>28 字节 | n bytes. Note: Each BOOL variable is one bit.<br>n 字节。注：一个 BOOL 量占一位。 |

Table 3-4 Frame header

表 3-4 帧头

| Field<br>数据域 | Type<br>类型 | Length<br>长度 | Value<br>数值 | Description<br>描述 |
|---|---|---|---|---|
| Port type | uint8_t | 1 byte | 1 | Port type: 0-Modbus master; 1-Modbus slave<br>端口类型：0-Modbus 主站；1-Modbus 从站 |
| Reserve 1 | uint8_t | 1 byte | | Reserve 1<br>预留 1 |
| Data length | uint16_t | 2 bytes | n | Modbus data length (bytes)<br>Modbus 数据的长度（字节） |
| Reserve 2 | uint8_t[] | 4 bytes | | Reserve 2<br>预留 2 |
| Section ID | uint32_t | 4 bytes | | Modbus data section ID<br>Modbus 数据区号 |
| Begin addr | uint16_t | 2 bytes | | Begin address of modbus variable - offset in section<br>Modbus 变量的起始地址 - Modbus 区内偏移地址 |
| Quantity of coils/No. of registers | uint16_t | 2 bytes | | Quantity of coils/ No. of registers<br>线圈/寄存器数 |
| Reserve 3 | uint8_t[] | 12 bytes | | Reserve 3<br>预留 3 |

### 3.2.6 Master read/write sample 主站读/写示例

#### 3.2.6.1 Master write sample 主站写示例

Master write sample is shown below: Modbus RTU protocol, little-endian order.

主站写操作示例如下图所示：Modbus RTU 协议，小端字节序。

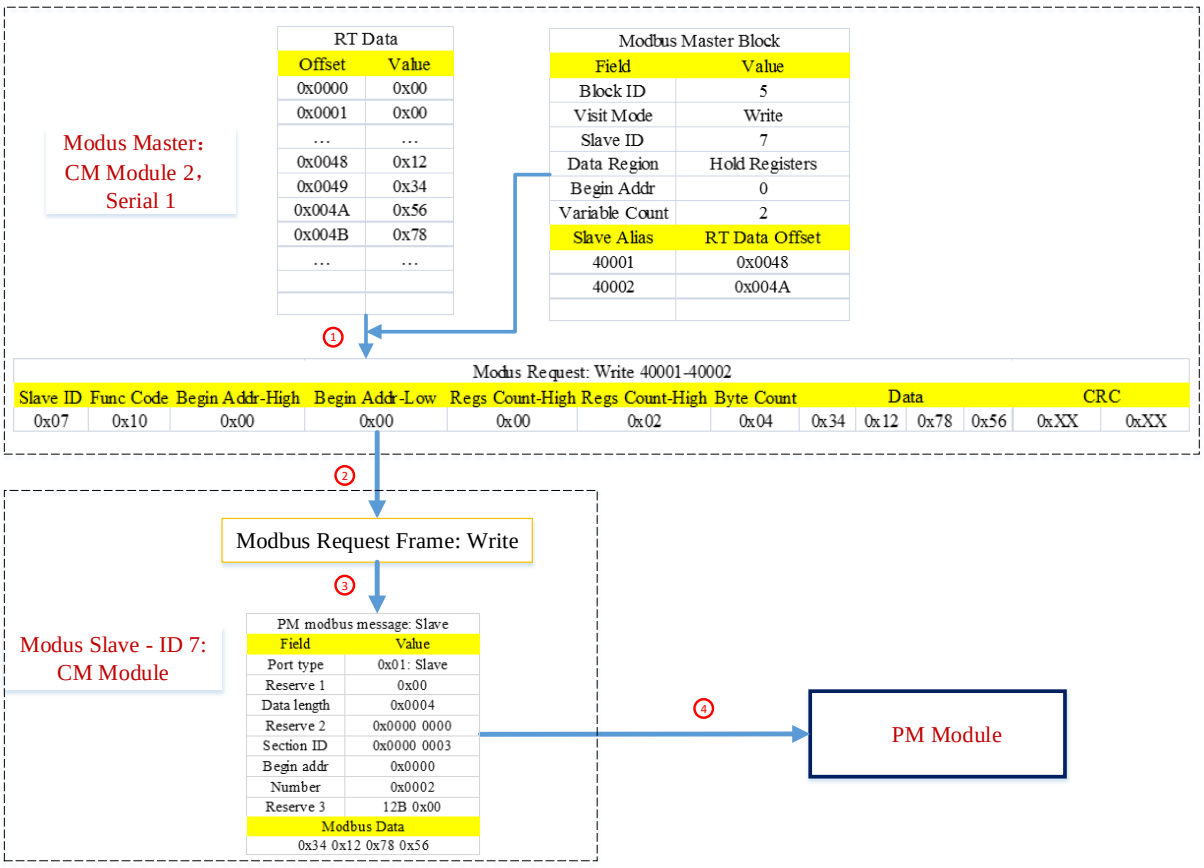


Figure 3-5 master write sample

图 3-5 主站写示例

### 3.2.6.2 Master read sample 主站读示例

Master read sample is shown below: Modbus RTU protocol, little-endian order.

主站读操作示例如下图所示：Modbus RTU 协议，小端字节序。

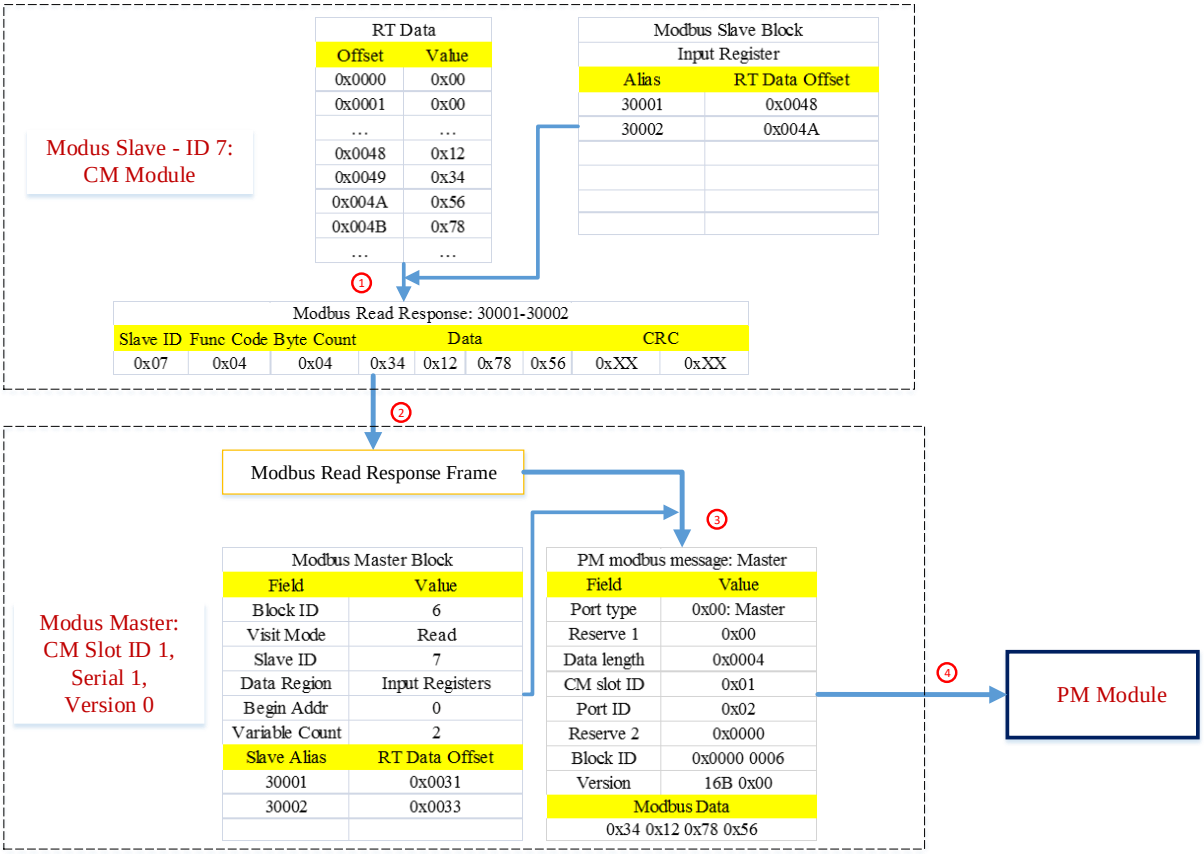


Figure 3-6 master read sample

图 3-6 主站读示例

## 3.3 Interface function 接口函数

The interface functions which are provided by this module are shown as follows:

模块提供的接口函数如下：

1. void ModbusInit(void)

| Input argument<br>输入参数 | Output argument<br>输出参数 | Description<br>描述 |
|---|---|---|
| No.<br>无。 | No.<br>无。 | Module initialization.<br>模块初始化。 |

2. void ModbusCycle(void)

| Input argument<br>输入参数 | Output argument<br>输出参数 | Description<br>描述 |
|---|---|---|
| No.<br>无。 | No.<br>无。 | Module cycle function.<br>模块周期运行函数。 |

## 3.4 Global variable 全局变量

Table 3-5 Global variable list

表 3-5 全局变量列表

| No.<br>序号 | Type<br>变量类型 | Name<br>名称 | Description<br>描述 |
|---|---|---|---|
| 1. | static bool_t | s_bCMHasCfg | CM has configuration flag.<br>CM 是否有配置标志。 |
| 2. | static uint8_t | s_ucLocalCMSlotID | Local CM slot ID.<br>本 CM 模块的槽位号。 |
| 3. | static uint8_t | s_ucMBTCPMstrIndex | Index of the current polling modbus TCP master station.<br>当前正在查询的 Modbus TCP 主站的索引值。 |
| 4. | static PortMstrInfo_t | s_stPortMstrInfo | Master port information.<br>主站端口信息。 |
| 5. | static SerlCfgInfo_t | s_stSerlCfgInfo | Serial configuration information.<br>串口配置参数。 |
| 6. | static MBTCPCfgInfo_t | s_stMBTCPInfo | Modbus TCP configuration information.<br>Modbus TCP 配置信息。 |
| 7. | static MBSlaveCfgInfo_t | s_stMBSlaveCfgInfo | Modbus slave configuration information.<br>Modbus 从站配置信息。 |
| 8. | static ModbusMstrCfgInfo_t | s_stMBMstrCfgInfo | Modbus master configuration information.<br>Modbus 主站配置信息。 |

## 3.5 Data structure 数据结构

1. Port master information structure

```
typedef struct PortMstrInfoTag
{
    ModuleState_t emState;
    bool_t bHandleSwitch;
    uint8_t ucReserve;
    uint16_t usEstimateRespLen;
    uint32_t uiBlockIndex;
    uint32_t uiWaitRespCycleCnt;
    uint32_t uiWaitLoopCycleCnt;
    ModbusMsg_t stMBReqMsg;
```

```
}PortMstrInfo_t;
```

2. Serial configure information structure

```
typedef struct SerlCfgInfoTag
{
    uint8_t ucIndex;
    uint8_t ucReserve[3];
    uint32_t uiCMCfgCRC;
    uint32_t uiCMCfgVer;
    RTDataVerInfo_t stRTDataVer;
    COMConfigInfo_t stCOMConfig[LOCAL_CFG_VER_CNT];
}SerlCfgInfo_t;
```

3. Modbus TCP configure information structure

```
typedef struct MBTCPCfgInfoTag
{
    uint8_t ucIndex;
    uint8_t ucReserve[3];
    uint32_t uiCMCfgCRC;
    uint32_t uiCMCfgVer;
    RTDataVerInfo_t stRTDataVer;
    ModbusTCPConfigInfo_t stMBTCPConfig[LOCAL_CFG_VER_CNT];
}MBTCPCfgInfo_t;
```

4. Modbus Slave configuration information structure

```
typedef struct MBSlaveCfgInfoTag
{
    uint8_t ucIndex;
    bool_t bCopying;
    bool_t bValid;
    uint8_t ucReserve;
    uint32_t uiCfgCRC;
    uint32_t uiCfgVer;
    RTDataVerInfo_t stRTDataVer;
    ModbusSlaveInfo_t stSlaveCfg[LOCAL_CFG_VER_CNT];
}MBSlaveCfgInfo_t;
```

5. Modbus master configuration information structure

```
typedef struct ModbusMasterCfgInfoTag
{
    uint8_t ucIndex;
```

```
    bool_t bCopying;
    bool_t bValid;
    uint8_t ucReserve;
    uint32_t uiCfgCRC;
    uint32_t uiCfgVer;
    RTDataVerInfo_t stRTDataVer;
    ModbusMasterInfo_t stMasterCfg[LOCAL_CFG_VER_CNT];
}ModbusMstrCfgInfo_t;
```

## 3.6 List of sub-function 子功能列表

The sub-functions list is shown as follows:

子功能列表如下：

Table 3-6 Sub function list

表 3-6 子功能列表

| Sub function No.<br>子功能编号 | Function description<br>功能描述 |
|---|---|
| SWDD-CM-MM_NSafR_SecR_A_001 | Module initialization<br>模块初始化 |
| SWDD-CM-MM_NSafR_SecR_A_002 | Module cycle function<br>模块周期运行函数 |

# 4 Design of sub-function 子功能设计

## 4.1 Module initialization 模块初始化

SWDD-CM-MM_NSafR_SecR_A_001

### 4.1.1 ModbusInit

#### 4.1.1.1 Function Description 功能描述

This function completes initialization of module.

该函数完成 Modbus 模块的初始化。

#### 4.1.1.2 Argument Description 参数说明

- Definition 函数定义

void ModbusInit(void)

- Input argument 输入参数

No.

无。

- Output argument 输出参数

No.

无。

#### 4.1.1.3 Processing flow 处理流程

This function has no branch and the processing flow is omitted.

此函数无分支，流程图省略。

## 4.2 Module cycle running function 模块周期运行函数

SWDD-CM-MM_NSafR_SecR_A_002

### 4.2.1 ModbusCycle

#### 4.2.1.1 Function Description 功能描述

This function completes module cycle running functions: manage configuration, realize modbus master and slave station port functions.

该函数完成模块周期运行功能：管理配置信息，实现 Modbus 主站和从站端口功能。

#### 4.2.1.2 Argument Description 参数说明

- Definition 函数定义

void ModbusCycle(void)

- Input argument 输入参数

No.

无。

- Output argument 输出参数

No.

无。

#### 4.2.1.3 Processing flow 处理流程

The processing flow is shown below, the main steps are as follows:

流程如下图所示，主要步骤如下：

1. Check if CM has configuration: if has, enter step 2, otherwise enter step 5;

   检查 CM 是否有配置：如果有，则进入步骤 2；否则进入步骤 5；

2. Handle configuration: see section 4.2.2 for details;

   管理配置：详见 4.2.2 节；

3. Handle modbus TCP: see section 4.2.3 for details;

   维护状态：详见 4.2.3 节；

4. Handle modbus serial: see section 4.2.4 for details;

   接收连接请求：详见 4.2.4 节；

5. Clear local information.

   清除本地信息。

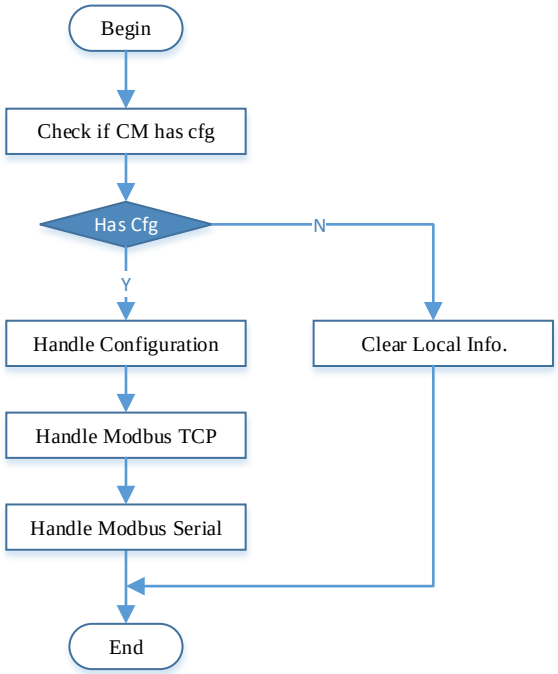


Figure 4-1 cycle running function

图 4-1 周期运行函数

### 4.2.2 HandleConfiguration

#### 4.2.2.1 Function Description 功能描述

This function is used to manage module configuration cyclically.

该函数用于周期性管理与本模块相关的配置信息。

#### 4.2.2.2 Argument Description 参数说明

- Definition 函数定义

static void HandleConfiguration(void)

- Input argument 输入参数

No.

无。

➢ Output argument 输出参数

No.

无。

#### 4.2.2.3 Processing flow 处理流程

The processing flow is shown below, the main steps are as follows:

流程如下图所示，主要步骤如下：

1. Handle modbus slave configuration;

   处理 Modbus 从站配置信息；

2. Handle modbus master configuration: if switch, enter step 3, otherwise end;

   处理 Modbus 主站配置：如果切换，则进入步骤 3，否则结束；

3. Check if master configuration of the specific port changed: if yes, update local master information of the specific port, otherwise check the next port.

   检查端口的主站配置是否发生了变化：如果是，则更新相应端口的本地主站信息，否则检查下一端口。

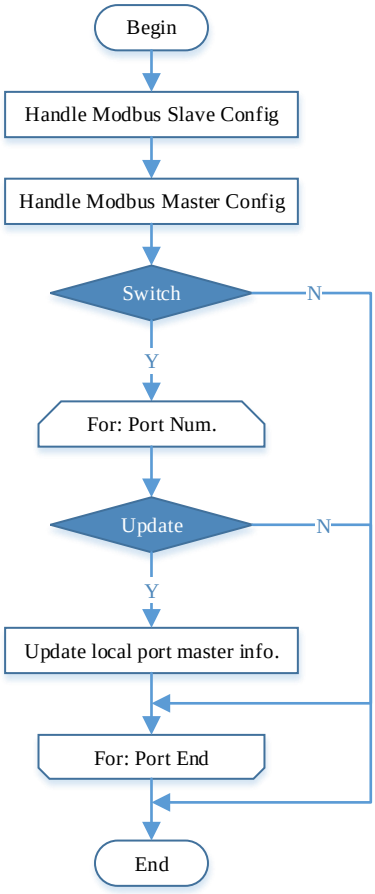


Figure 4-2 handle configuration

图 4-2 处理配置

### 4.2.3 HandleModbusTCP

#### 4.2.3.1 Function Description 功能描述

This function is used to handle modbus TCP configuration of the network port, and handle modbus TCP message according to the configuration cyclically.

该函数用于周期性处理网口的 modbus TCP 配置信息，并根据配置信息处理 Modbus TCP 消息。

#### 4.2.3.2 Argument Description 参数说明

- Definition 函数定义

static void HandleModbusTCP(void)

- Input argument 输入参数

No.

无。

- Output argument 输出参数

No.

无。

#### 4.2.3.3 Processing flow 处理流程

The processing flow is shown below, the main steps are as follows:

流程如下图所示，主要步骤如下：

1. Check if can handle port configuration: if yes, enter step 2, otherwise enter step 3;

   检查能否处理端口配置：如果能，则进入步骤 2；否则进入步骤 3；

2. Handle port configuration: If changed, copy to local and switch, and then update local port master information;

   处理端口配置：如果发生变化，则拷贝到本地并切换，然后更新本地端口主站信息；

3. Check if port configuration is actived & modbus protocol: if yes, enter step 4, otherwise check the next network port;

   检查端口配置是否已被激活且端口协议为 Modbus：如果是，则进入步骤 4，否则检查下一网络端口；

4. Check port protocol: if master, enter step 5; if slave, enter step 6; if all, enter step 7;

   检查端口协议：如果主站，则进入步骤 5；如果从站，则进入步骤 6；如果主从站，

则进入步骤 7；

5. Handle modbus TCP master: see section 4.2.5 for details;

   处理 modbus TCP 主站端口功能：详见 4.2.5 节；

6. Handle modbus TCP slave: see section 4.2.6 for details;

   处理 modbus TCP 从站端口功能：详见 4.2.6 节；

7. Handle modbus TCP master & slave: see section 4.2.5 & 4.2.6 for details.

   处理 modbus TCP 主从站端口功能：详见 4.2.5 和 4.2.6 节。

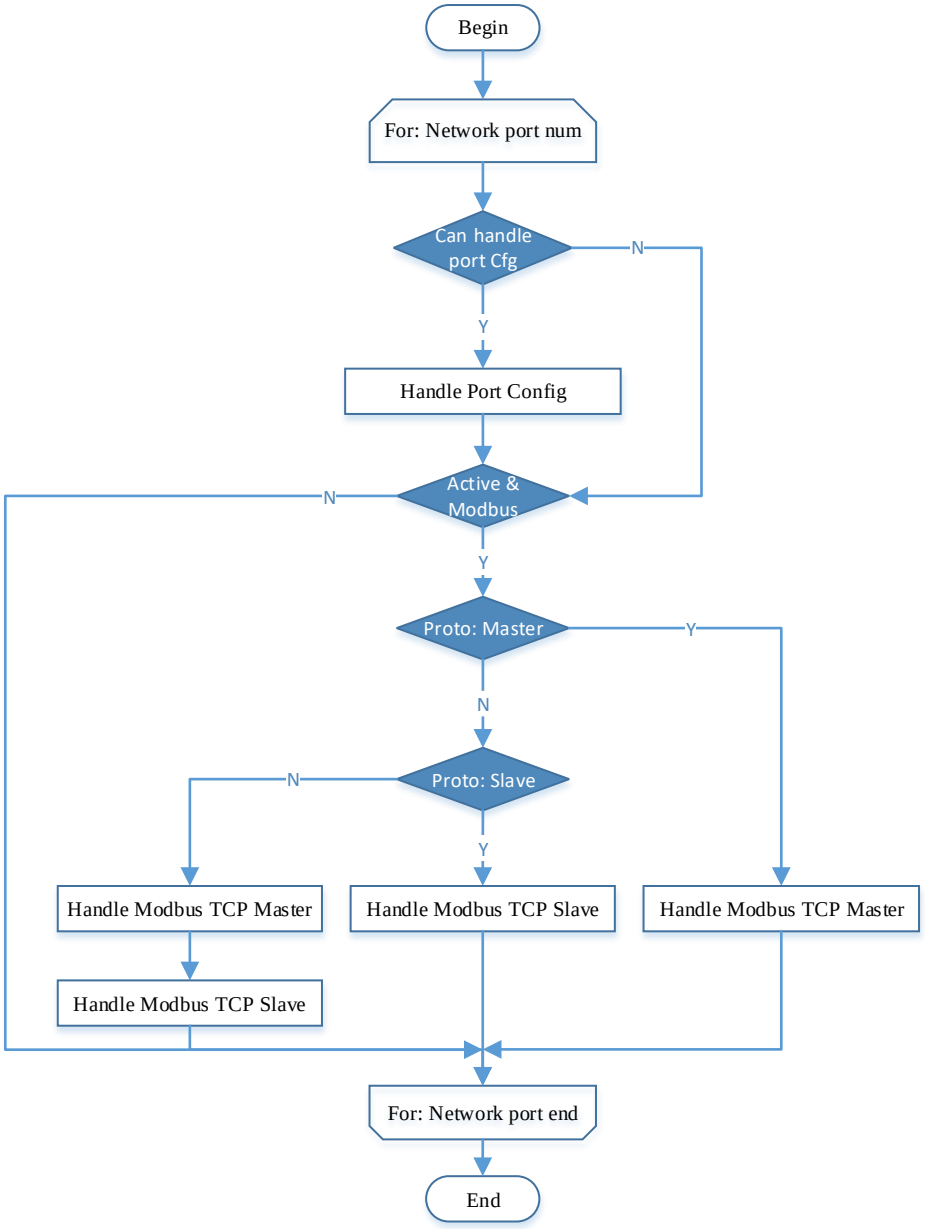


Figure 4-3 handle modbus TCP function

图 4-3 处理 modbus TCP 功能

### 4.2.4 HandleModbusSerial

#### 4.2.4.1 Function Description 功能描述

This function is used to handle serial port configuration, and handle modbus serial message according to the configuration cyclically.

该函数用于周期性处理串口配置信息，并根据配置信息处理 Modbus 串口消息。

#### 4.2.4.2 Argument Description 参数说明

- Definition 函数定义

static void HandleModbusSerial(void)

- Input argument 输入参数

No.

无。

- Output argument 输出参数

No.

无。

#### 4.2.4.3 Processing flow 处理流程

The processing flow is shown below, the main steps are as follows:

流程如下图所示，主要步骤如下：

1. Check if can handle port configuration: if yes, enter step 2, otherwise enter step 3;

   检查能否处理端口配置：如果能，则进入步骤 2；否则进入步骤 3；

2. Handle port configuration: If changed, copy to local and switch, and then update local port master information;

   处理端口配置：如果发生变化，则拷贝到本地并切换，然后更新本地端口主站信息；

3. Check if port configuration is actived & modbus protocol: if yes, enter step 4, otherwise check the next serial port;

   检查端口配置是否已被激活且端口协议为 Modbus：如果是，则进入步骤 4，否则检查下一串口；

4. Check port protocol: if master, enter step 5, otherwise enter step 6;

   检查端口协议：如果主站，则进入步骤 5，否则进入步骤 6；

5. Handle modbus serial master: see section 4.2.5 for reference;

   处理 modbus 串口主站功能：参见 4.2.5 节；

6. Handle modbus serial slave: see section 4.2.6 for reference.

处理 modbus 串口从站功能：参见 4.2.6 节。

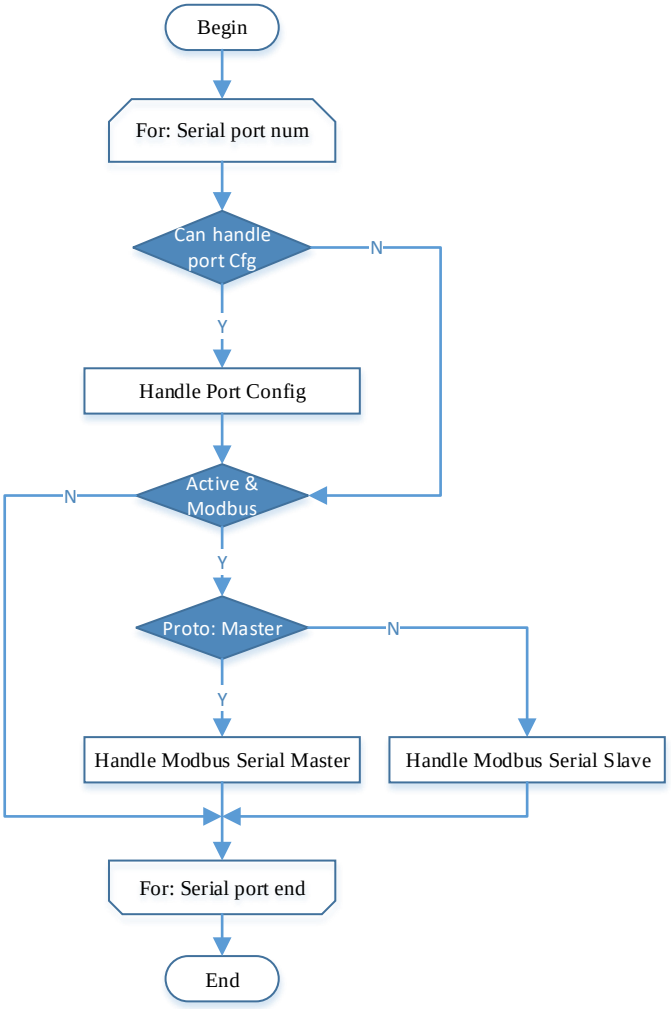


Figure 4-4 handle modbus serial function

图 4-4 处理 modbus 串口功能

### 4.2.5 HandleModbusTCPMaster

#### 4.2.5.1 Function Description 功能描述

This function is used to poll modbus slave station(s) according to the configuration.

该函数用于根据配置信息轮询相应的 modbus 从站。

#### 4.2.5.2 Argument Description 参数说明

- Definition 函数定义

static void HandleModbusTCPMaster(CommPort_t emPort)

- Input argument 输入参数

emPort: Port ID 端口 ID.

➢ Output argument 输出参数

No.

无。

#### 4.2.5.3 Processing flow 处理流程

The processing flow is shown below, the main steps are as follows:

流程如下图所示，主要步骤如下：

1. Update wait cycle count: loop next slave station when it reaches to 0;

   更新等待周期计数值：数值为 0 时，轮询下一从站；

2. Check master port status: if idle, enter step 3, otherwise enter step 4;

   检查主站端口状态：如果空闲，则进入步骤 3，否则进入步骤 4；

3. Handle idle status: make request message and send to the corresponding slave station, see section 4.2.7 for details;

   处理空闲状态：构造请求帧并发送给相应的从站，详见 4.2.7 节；

4. Handle busy status: handle response of the corresponding slave station, see section 4.2.8 for details.

   处理忙状态：处理相应从站的应答，详见 4.2.8 节。

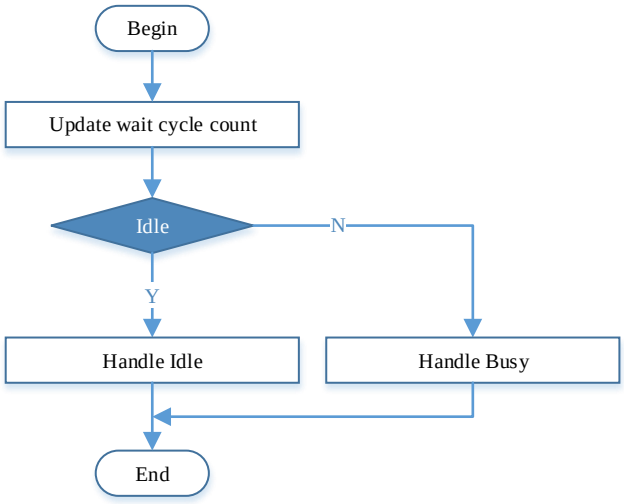


Figure 4-5 handle modbus TCP function

图 4-5 处理 modbus TCP 功能

### 4.2.6 HandleModbusTCPSlave

#### 4.2.6.1 Function Description 功能描述

This function is used to handle request message of the modbus master station, and response to it.

该函数用于处理 modbus 主站请求并应答。

#### 4.2.6.2 Argument Description 参数说明

- Definition 函数定义

static void HandleModbusTCPSlave(void)

- Input argument 输入参数

No.

无。

- Output argument 输出参数

No.

无。

#### 4.2.6.3 Processing flow 处理流程

The processing flow is shown below, the main steps are as follows:

流程如下图所示，主要步骤如下：

1. Get a request message from the current modbus TCP master station;

   获取来自当前 modbus TCP 主站的请求帧；

2. Check if get: if yes, enter step 4, otherwise enter step 3;

   检查是否获取到：如果是，则进入步骤 4，否则进入步骤 3；

3. Update index of the modbus TCP master station, and check if continue: if yes, enter step 1, otherwise end;

   更新 modbus TCP 主站索引值，检查是否应继续处理，如果是，则进入步骤 1，否则结束；

4. Decode MBTCP request message: see section 4.2.9 for details;

   解析来自 modbus TCP 主站的请求消息：详见 4.2.9 节；

5. Check if has response message: if has, send to the current modbus TCP master station, otherwise add handle count and enter step 1.

   检查是否有应答消息：如果有，则发送到当前 modbus TCP 主站，否则增加处理计数值并进入步骤 1。

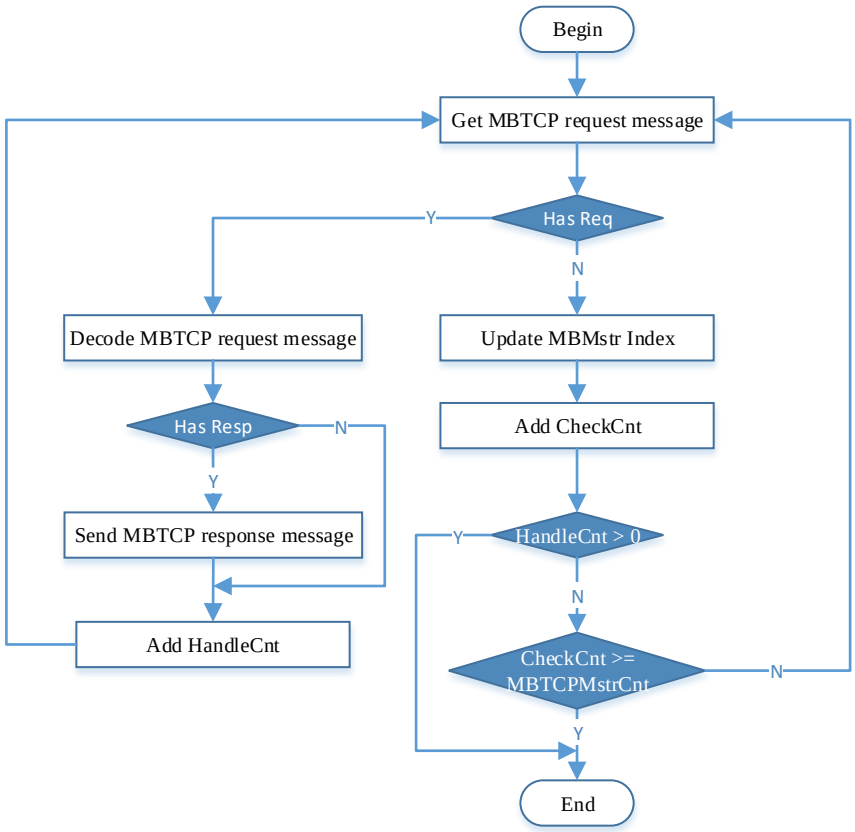


Figure 4-6 handle modbus TCP function

图 4-6 处理 modbus TCP 功能

### 4.2.7 HandleMBTCPMstrIdle

#### 4.2.7.1 Function Description 功能描述

This function is used to make request message and send to the corresponding slave station.

该函数用于构造请求帧并发送给相应的从站。

#### 4.2.7.2 Argument Description 参数说明

- Definition 函数定义

static void HandleMBTCPMstrIdle(CommPort_t emPort, PortMstrInfo_t *pstPortMstrInfo)

- Input argument 输入参数

emPort: Port ID 端口 ID；

pstPortMstrInfo: Port master information 端口主站信息。

- Output argument 输出参数

No.

无。

#### 4.2.7.3 Processing flow 处理流程

The processing flow is shown below:

流程如下图所示：

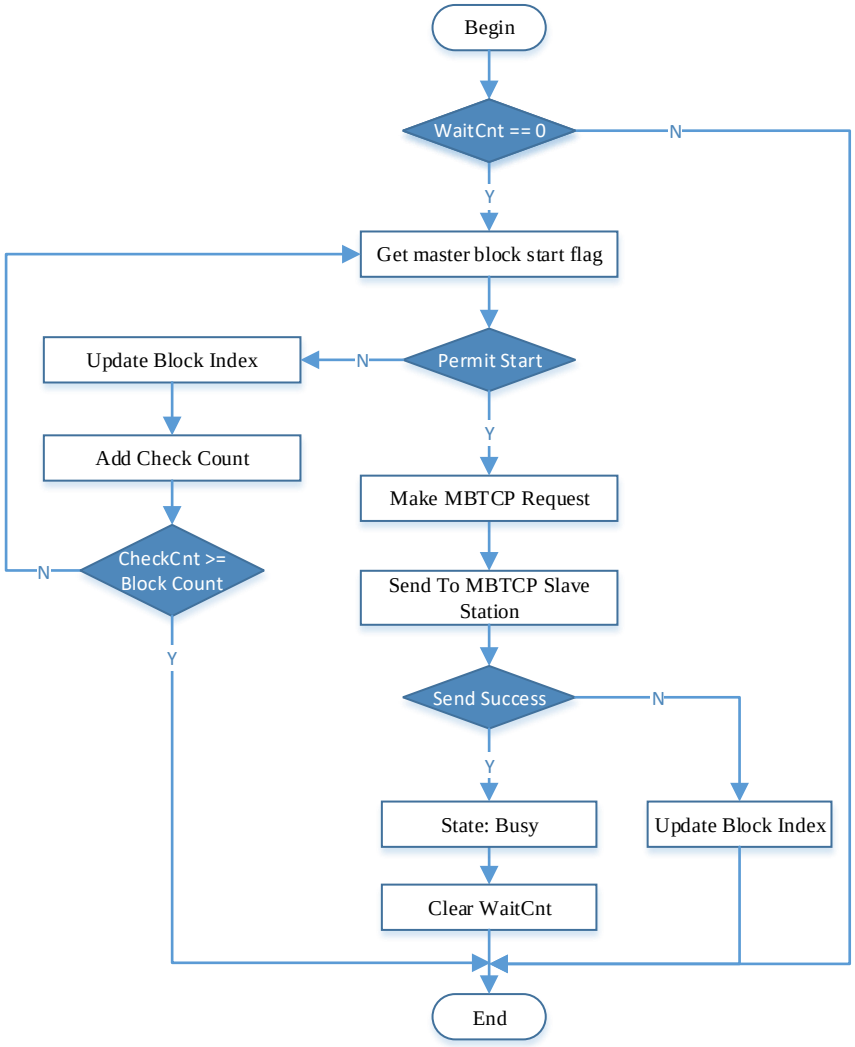


Figure 4-7 handle modbus TCP master idle function

图 4-7 处理 modbus TCP 主站空闲功能

### 4.2.8 HandleMBTCPMstrBusy

#### 4.2.8.1 Function Description 功能描述

This function is used to handle response of the corresponding slave station.

该函数用于处理相应的从站的应答。

#### 4.2.8.2 Argument Description 参数说明

- Definition 函数定义

static void HandleMBTCPMstrBusy(CommPort_t emPort, PortMstrInfo_t *pstPortMstrInfo)

➢ Input argument 输入参数

emPort: Port ID 端口 ID；

pstPortMstrInfo: Port master information 端口主站信息。

➢ Output argument 输出参数

No.

无。

#### 4.2.8.3 Processing flow 处理流程

The processing flow is shown below:

流程如下图所示：

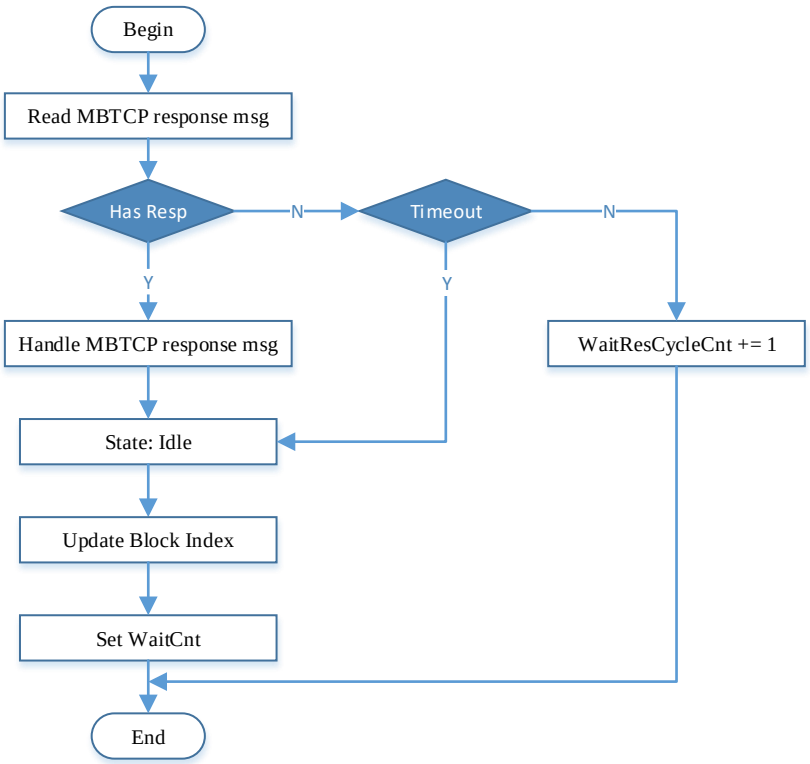


Figure 4-8 handle modbus TCP master busy function

图 4-8 处理 modbus TCP 主站忙功能

### 4.2.9 DecodeModbusTCPReq

#### 4.2.9.1 Function Description 功能描述

This function is used to decode request message, and make response message according to the decode result. When the request message is valid and write operation, send data to PM.

该函数用于解析请求帧，并根据解析结果构造应答帧。当请求帧有效且为写操作时，将数据发送到 PM。

#### 4.2.9.2 Argument Description 参数说明

- Definition 函数定义

static void DecodeModbusTCPReq(CommPort_t emPort, uint8_t const pucReq[], uint16_t usReqLen, uint8_t pucResp[], puint16_t pusRespLen)

- Input argument 输入参数

emPort: Port ID 端口 ID；

pucReq: Request message address 请求消息地址；

usReqLen: Request message length 请求消息长度；

pucResp: Response message address 应答消息地址；

- Output argument 输出参数

pusRespLen: Response message length 应答消息长度。

#### 4.2.9.3 Processing flow 处理流程

The processing flow is shown below:

流程如下图所示：

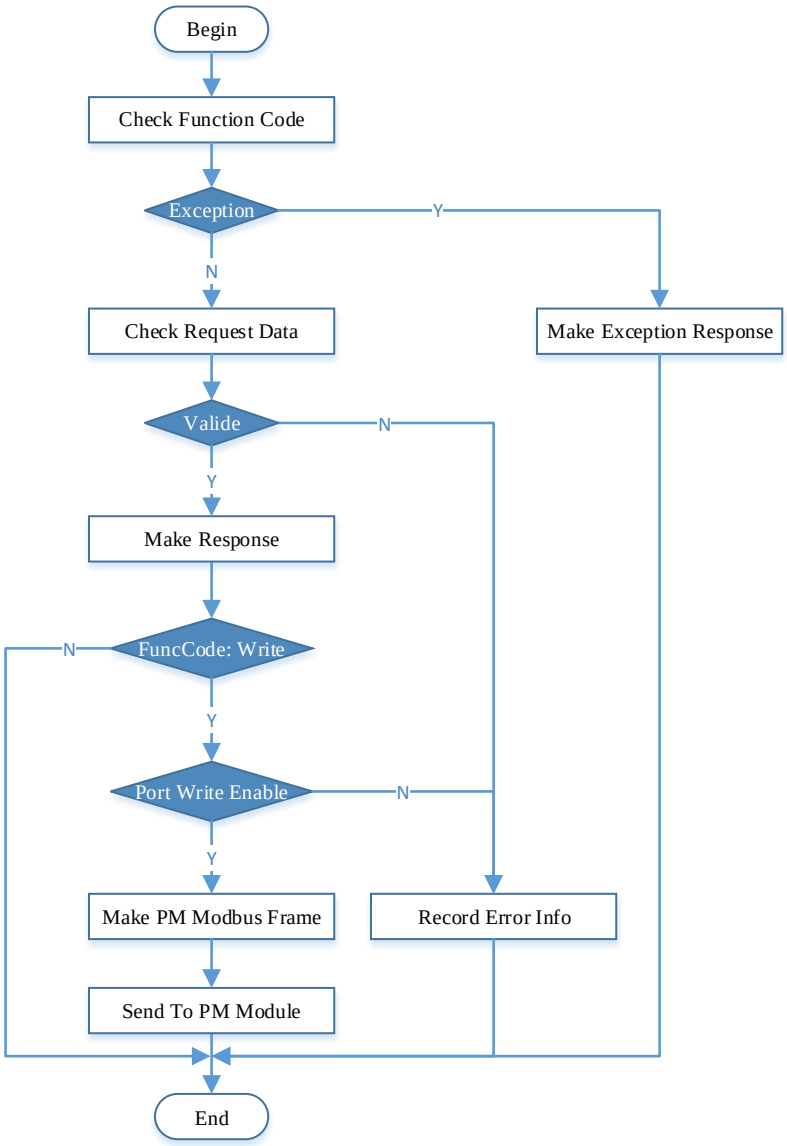


Figure 4-9 decode modbus TCP master request message

图 4-9 解析 modbus TCP 主站请求消息

——以下无正文