Document Title: CM_FW system resource management module design description of Safety Control System

Document Number: 16-Q04-000128

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: Li Qi 2016-12-26

Checked by: Zhu Genghua 2016-12-30

Countersigned by: Liu Yang, Wang Dong

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	2016-12-26	None	A	Li Qi	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 系统资源管理模块设计说明书

文件编号：16-Q04-000128

项目编号：CT-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：李琦 2016 年 12 月 26 日

审核：朱耿华 2016 年 12 月 30 日

会签：刘阳、王东

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	2016-12-30		A	李琦	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 system resource management 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 系统资源管理模块的设计方案。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. 序号	Term 术语	Description 解释
1.	IP_BUS	Communication between PM and IO modules. PM 与 IO 模块之间的通讯总线。
2.	CM_BUS	Communication between PM and CM. PM 与 CM 之间的通讯总线。
3.	PM_BUS	Communication between PMs. PM 之间的通讯总线。
4.	System Net	Communication between control station and PC. 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 控制站	A set of triple redundant control system, which includes triple redundant PMs and IO modules under control. 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 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. 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 控制周期	Time interval between adjacent two runs of user program execution. PM 两次执行用户程序间隔时间。
9.	Project 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. 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 源工程文件	Source file of the project before compiling. 工程在编译前的源文件。
11.	User program 用户程序	Part of project which contain user control program (binaries) to be loaded and executed and configuration information of task. 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 缩略语

Table 1-2 Abbreviations

表 1-2 缩略语

No. 序号	Abbreviation 缩略语	English description 英文	Chinese description 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 system resource management module (marked red) in the software hierarchy is shown below.

系统资源管理模块（标红）在软件层次中的位置如下图所示。

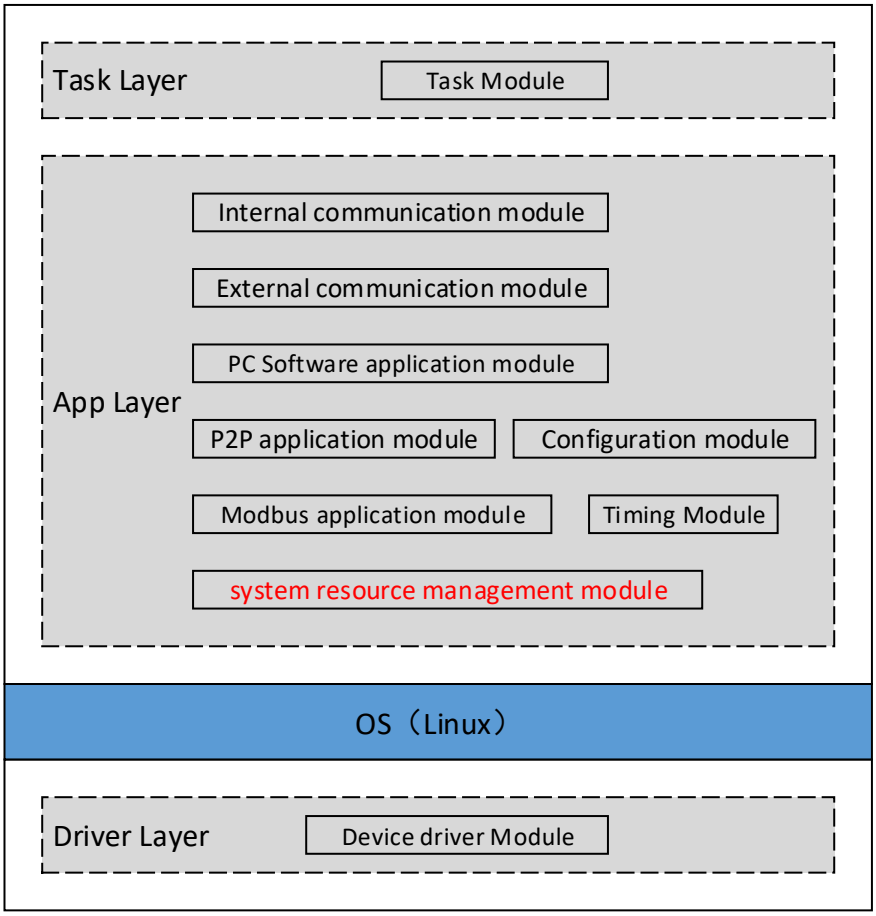

Figure 2-1 the location of the system resource module

图 2-1 模块位置

The system resource management module defines massive set and get interface, and other modules

access the global variables via set/get interface.

系统资源模块中定义了大量的 set 和 get 接口函数，其它模块通过这些接口函数实现对全局变量的访问操作。

The module is also responsible for setting the status of the LED lights, providing the CM version and providing common library functions.

模块同时负责设置 LED 灯的状态、提供 CM 的版本及提供公共库函数

3 Module design 模块设计

3.1 Function description 功能描述

1) Common 通用状态

Set or get CM/PM's state, set or get system time, set or get internal communication state, set or get external communication state, set or get CM's slot ID, set or get CM's logic ID, set or get project information, set or get user task number, set or get the information which will be transferred from CM to PM.

设置或获取 CM/PM 系统状态、设置或获取系统时间、设置或获取内部通讯状态、设置或获取外部通讯状态、设置或获取 CM 槽号、设置或获取 CM 逻辑号、设置或获取工程信息、设置或获取 PM 中用户任务个数、设置 CM 需要传输到 PM 的各种状态信息。

2) Configuration 配置信息

Set or get related configuration.

设置或获取各种配置信息

3) Modbus

Read or write Modbus/Modbus message.

读写 Modbus/Modbus 消息

4) P2P

Read or write P2P message.

读写 P2P 消息

5) PC software 上位机软件

Read or write PC software information.

读写上位机软件消息

6) Real time data 实时数据

Read real time data and send response.

读取实时数据，发送应答

7) Configuration file 配置文件

Read or write configuration file.

读写配置文件

8) FPGA registers FPGA 寄存器

Read or write FPGA register interface.

读写 FPGA 相关接口函数

9) Common library 公共库函数

Calculate CRC32 and CRC16 functions are provided.

计算 CRC32 和 CRC16 的函数。

10) LED 状态灯

Get or set LED's state and update LED's state.

获取或设置 LED 灯的状态以及更新 LED 灯的状态。

11) Version 版本号

Show the version of CM.

显示 CM 的版本号

3.2 Design concept 设计思路

The system resource management module defined massive 'set' and 'get' interfaces, and other modules access the system resource (static variables) via set/get interfaces. The system resource access data flow is figure3-1.

系统资源管理模块中定义了大量的"set"和"get"接口函数，其它模块通过这些接口函数实现对系统资源的访问操作。系统资源访问数据流图如图 3-1 所示。

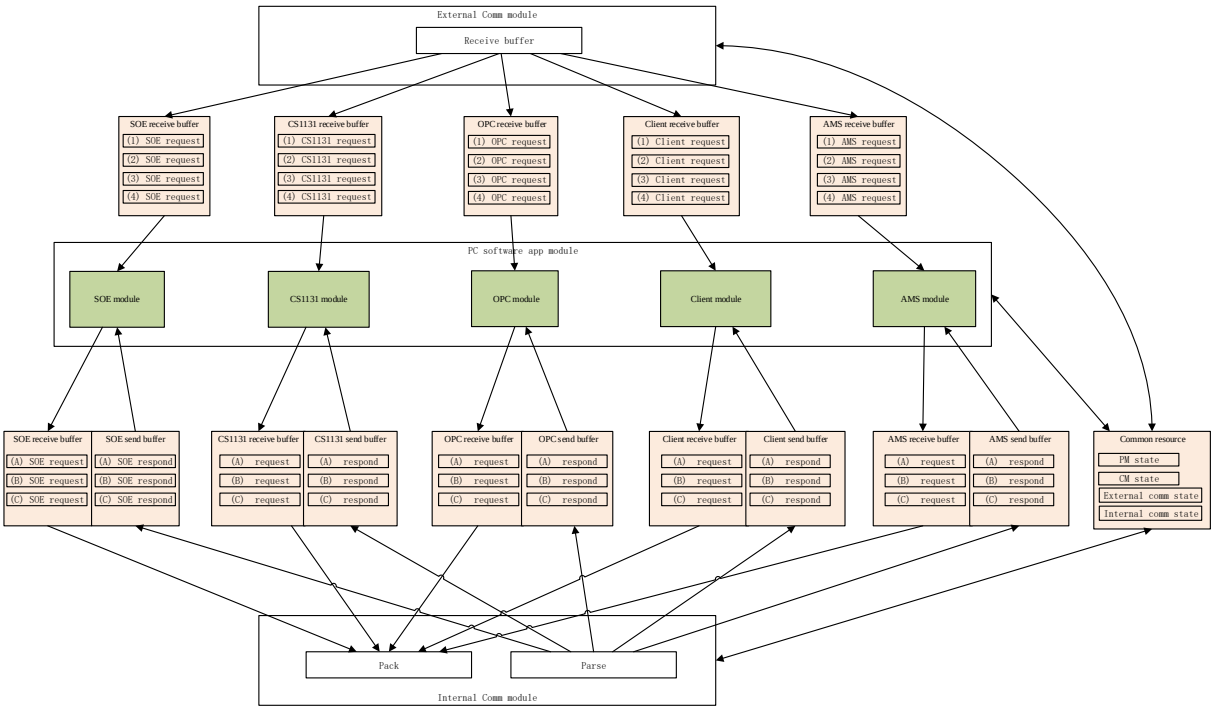

Figure3-1 system resource access data flow

图 3-1 系统资源访问数据流图

3.3 Interface function 接口函数

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

模块提供的接口函数如下：

1. void SysInit (void)

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

2. bool_t SysGetPMState(PMController_t emPMID, uint16_t *pusState);

Input argument 输入参数	Output argument 输出参数	Description 描述
emPMID. PM's ID PM 的 ID 号。	pusState. PM's state PM 的状态。	Get the related PM's state 获取相关 PM 的状态

3. bool_t SysSetPMState(PMController_t emPMID, uint16_t usPMState);

Input argument 输入参数	Output argument 输出参数	Description 描述

emPMID. PM's ID PM 的 ID 号。 usState. PM's state PM 的状态。	No. 无	Set the related PM's state 设置相关 PM 的状态

4. bool_t SysGetCMState(CMController_t emCMID, uint16_t *pusState);

Input argument 输入参数	Output argument 输出参数	Description 描述
emCMID. CM's ID CM 的 ID 号。	pusState. CM's state CM 的状态。	Get the related CM's state 获取相关 CM 的状态

5. bool_t SysSetCMState(CMController_t emCMID, uint16_t usCMState);

Input argument 输入参数	Output argument 输出参数	Description 描述
emCMID. PM 的 id 号。 PM's ID usCMState. CM's state CM 的状态。	No. 无	Set the related CM's state 设置相关 CM 的状态

6. bool_t SysSetSysState(SysState_t *pstSysState, PMController_t emPMID);

Input argument 输入参数	Output argument 输出参数	Description 描述
pstSysState. Point to system state 指向系统状态信息。 emPMID. PM's ID PM 的 ID 号。	No. 无	Set system state 设置系统状态

7. bool_t SysGetSysTimeFromPM(SysSystemTime_t *pstSysTime, PMController_t emPMID);

Input argument 输入参数	Output argument 输出参数	Description 描述

emPMID. PM's ID PM 的 ID 号。	pstSysTime point to system time from PM 输出来自 PM 的系统时间	Get system time from PM 获取来自 PM 的系统时间

8. uint16_t SysGetInterCommState(CMController_t emCMID, PMController_t emPMID);

Input argument 输入参数	Output argument 输出参数	Description 描述
emCMID. CM's ID CM 的 ID 号。 emPMID. PM's ID PM 的 ID 号。	No 无	Get internal communication state between CM and PM 获取 PM 与 CM 间的内部通讯状态

9. bool_t SysSetInterCommState(CMController_t emCMID, PMController_t emPMID, uint16_t usInterState);

Input argument 输入参数	Output argument 输出参数	Description 描述
emCMID. CM's ID CM 的 ID 号。 emPMID. PM's ID PM 的 ID 号。 usInterState Internal communication state	No 无	Set internal communication state between CM and PM 设置 PM 与 CM 间的内部通讯状态

10. SysSystemTime_t SysGetCMSystemTime(CMController_t emCMID);

Input argument 输入参数	Output argument 输出参数	Description 描述
emCMID. CM's ID CM 的 ID 号。	No 无	Get system time in CM 获取 CM 中的系统时间

11. bool_t SysSetCMSystemTime(CMController_t emCMID, SysSystemTime_t *pstSysTime);

Input argument 输入参数	Output argument 输出参数	Description 描述
emCMID. CM's ID CM 的 ID 号。 pstSysTime point to system time 指向系统时间	 No 无	 Set system time in CM 设置 CM 中的系统时间

12. CMController_t SysGetLocalCMID(void);

Input argument 输入参数	Output argument 输出参数	Description 描述
No 无	No 无	Get CM's ID 获取 CM 的 ID 号

13. CMSlot_t SysGetLocalCMSlot (void);

Input argument 输入参数	Output argument 输出参数	Description 描述
No 无	No 无	Get CM's slot 获取 CM 的槽号

14. uint8_t SysGetPMTaskNum(PMController_t emPMID);

Input argument 输入参数	Output argument 输出参数	Description 描述
emPMID. PM's ID PM 的 ID 号。	No 无	Get user task number in PM 获取 PM 中用户任务的个数

15. void SysConfigInit(void);

Input argument 输入参数	Output argument 输出参数	Description 描述
No 无	No 无	Configuration initialization 配置信息初始化

16. bool_t SysGetAccCtrlInfo(AccessCtrlConfigInfo_t *pstAccCtrlInfo);

Input argument 输入参数	Output argument 输出参数	Description 描述
No 无	pstAccCtrlInfo point to access control	Get access control information 获取访问控制信息

	information 指向访问控制信息	

17. bool_t SysGetModbusTcpConfigInfo(NetworkPortType_t emType, ModbusTCPConfigInfo_t *pstModbusTcpConfigInfo);

Input argument 输入参数	Output argument 输出参数	Description 描述
emType Network type (NET1 or NET2)	pstModbusTcpConfigInfo point to ModbusTCP configuration information 指向 ModbusTCP 配置信息	Get ModbusTCP configuration information 获取 ModbusTCP 配置信息

18. void SysModbusInit(void);

Input argument 输入参数	Output argument 输出参数	Description 描述
No 无	No 无	Modbus initialization Modbus 初始化

19. bool_t SysGetModbusMsgQueue (PMController_t emPMID, puint8_t pucContent, puint16_t pusLen);

Input argument 输入参数	Output argument 输出参数	Description 描述
emPMID PM's ID PM 的 ID 号	pucContent point to Modbus message content 指向 Modbus 消息内容 pusLen point to Modbus message length 指向 Modbus 消息长度	Get Modbus message from Modbus queue 从队列中获取 Modbus 消息

20. bool_t SysSetModbusMsgQueue (PMController_t emPMID, puint8_t pucContent, uint16_t usLen);

Input argument 输入参数	Output argument 输出参数	Description 描述
emPMID PM's ID	No 无	Set Modbus message to Modbus queue

PM 的 ID 号 pucContent point to Modbus message content 指向 Modbus 消息内容 usLen Modbus message length Modbus 消息长度		将 Modbus 消息写入队列

21. void SysP2PInit (void);

Input argument 输入参数	Output argument 输出参数	Description 描述
No 无	No 无	P2P initialization P2P 初始化

22. bool_t SysGetP2PMsg(PMController_t emPMID, SysP2PMsgType_t emMsgType, puint8_t pucContent, puint16_t pusLen);

Input argument 输入参数	Output argument 输出参数	Description 描述
emPMID PM's ID PM 的 ID 号 emMsgType Message type P2P 消息类型	pucContent point to P2P message content 指向 P2P 消息内容 pusLen point to P2P message length 指向 P2P 消息长度	Get P2P message 获取 P2P 消息

23. bool_t SysSetP2PMsg(PMController_t emPMID, SysP2PMsgType_t emMsgType , puint8_t pucContent, uint16_t usLen);

Input argument 输入参数	Output argument 输出参数	Description 描述
emPMID PM's ID PM 的 ID 号 emMsgType	No 无	Set P2P message 存储 P2P 消息

Message type P2P 消息类型 pucContent point to P2P message content 指向 P2P 消息内容 pusLen point to P2P message length 指向 P2P 消息长度		

24. void SysPCsoftwareInit (void);

Input argument 输入参数	Output argument 输出参数	Description 描述
No 无	No 无	PC software initialization 上位机软件初始化

25. bool_t SysGetMsg (PMController_t emPMID, SysMsgType_t emMsgType, puint8_t pucContent, puint16_t pusLen);

Input argument 输入参数	Output argument 输出参数	Description 描述
emPMID PM's ID PM 的 ID 号 emMsgType Message type 消息类型	pucContent point to message content 指向消息内容 pusLen point to message length 指向消息长度	Get message 获取消息

26. bool_t SysSetMsg (PMController_t emPMID, SysMsgType_t emMsgType , puint8_t pucContent, uint16_t usLen);

Input argument 输入参数	Output argument 输出参数	Description 描述
emPMID PM's ID PM 的 ID 号 emMsgType Message type 消息类型	No 无	Set message 存储消息

pucContent point to message content 指向消息内容 pusLen point to message length 指向消息长度		

27. void SysRTDataInit (void);

Input argument 输入参数	Output argument 输出参数	Description 描述
No 无	No 无	Real time data initialization 实时数据初始化

28. bool_t SysGetRTDataMsg (PMController_t emPMID, SysRTDataMsgType_t emMsgType, puint8_t pucContent, puint16_t pusLen);

Input argument 输入参数	Output argument 输出参数	Description 描述
emPMID PM's ID PM 的 ID 号 emMsgType Message type 消息类型	pucContent point to real time data content 指向实时数据内容 pusLen point to real time data length 指向实时数据长度	Get real time data 获取实时数据

29. bool_t SysSetRTDataMsg (PMController_t emPMID, SysRTDataMsgType_t emMsgType , puint8_t pucContent, uint16_t usLen);

Input argument 输入参数	Output argument 输出参数	Description 描述
emPMID PM's ID PM 的 ID 号 emMsgType Message type 消息类型	No 无	Set real time data to system resource module 写入实时数据的系统资源管理模块

pucContent point to message content 指向消息内容 pusLen point to message length 指向消息长度		

30. void SysDatabseInit (void);

Input argument 输入参数	Output argument 输出参数	Description 描述
No 无	No 无	Data base initialization 数据库初始化

31. void SysClearDatabase (void);

Input argument 输入参数	Output argument 输出参数	Description 描述
No 无	No 无	Clear data base 清除数据库

32. uint16_t SysGetCfgFileInfo (SysCfgFileType_t emCfgFileType, uint32_t uiRdOffset, puint8_t pucContent, uint16_t usLen);

Input argument 输入参数	Output argument 输出参数	Description 描述
emCfgFileType Configuration file type 配置文件类型 uiRdOffset read offset 读取偏移地址 usLen read data length 读取数据长度	pucContent point to file content 指向读取的文件内容	Get configuration file content 读取配置文件内容

33. uint16_t SysSetCfgFileInfo(SysCfgFileType_t emCfgFileType, puint8_t pucContent, uint16_t usLen);

Input argument 输入参数	Output argument 输出参数	Description 描述
emCfgFileType Configuration file type 配置文件类型 pucContent point to content 指向待写入的内容 usLen read data length 读取数据长度	No 无	Set configuration file content 写入配置文件内容

3.4 Global variable 全局变量

Table 3-1 Global variable list

表 3-1 全局变量列表

No. 序号	Type 变量类型	Name 名称	Description 描述
1.	SysState_t	s_stSysState	System state. 系统状态
2.	SysSystemTime_t	s_stSystemTime	System time 系统时间
3.	PrjInfo_t	s_stPrjInfo[NUM_OF_PM]	Project information 工程信息
4.	TaskInformation_t	s_stTaskInfo[NUM_OF_PM]	Task information 任务信息
5.	uint16_t	s_usInterCommState [NUM_OF_PM][NUM_OF_SLOT]	Internal communication state 内部通讯状态
6.	SysCM2PM_t	s_stCM2PMInfo	CM to PM information

			CM 到 PM 的信息
7.	RTDataVerInfo_t	s_stRTDataVer[NUM_OF_PM]	Real time data version 实时数据版本号
8.	SysModbusMsgQueue_t	s_stSysModbusMsgQueue [NUM_OF_PM]	Modbus message queue Modbus 消息队列
9.	SysModbusTCPMsgQueue_t	s_stSysModbusTcpMsgQueue [MAX_MODBUS_TCP_MASTER_NUM]	Modbus TCP message queue Modbus TCP 消息队列
10.	SysP2PMsgBox_t	s_stSysP2PMsgBox [NUM_OF_PM] [NUM_OF_P2PMSGTYPE]	P2P message box 存储 P2P 消息
11.	SysP2PMsgQueue_t	s_stSysP2PExtReqMsgQueue	Request message queue 请求消息队列
12.	SysP2PMsgQueue_t	s_stSysP2PExtRespMsgQueue	Response message queue 应答消息队列
13.	SysMsgBox_t	s_stSysMsgBox [NUM_OF_PM] [NUM_OF_MSGTYPE];	Message box 存储来自 PM 的消息
14.	SysAppMsgBox_t	s_stSysAppReqMsgBox [NUM_OF_SOFTWARETYPE] [APP_SW_MAX_NUM]	Request message from PC software 来自 PC 软件的请求
15.	SysAppMsgBox_t	s_stSysAppRespMsgBox [NUM_OF_SOFTWARETYPE] [APP_SW_MAX_NUM]	Response message to PC software 发送到 PC 软件的应答
16.	SysRTDataMsgBox_t	s_stSysRTDataMsgBox	Real time data

		[NUM_OF_PM]	message box 存储实时数据
17.	SysRTDataAckMsgBox_t	s_stSysRTDataAckMsgBox [NUM_OF_PM]	Real time data ACK message box 存储应答消息
18.	RTDataArea_t	s_stRTDataArea [NUM_OF_PM] [NUM_OF_RTDATA_AREA]	Real time data area 实时数据区
19.	SysCfgDB_t	s_stSysCfgDB	Data base 数据库

3.5 Data structure 数据结构

1. PM ID PM 的 ID 号

```
typedef enum SysPMControllerTag
{
    PMA = 0x00,
    PMB,
    PMC,
    NUM_OF_PM,
    INVALID_PM_ID
} PMController_t;
```

2. CM ID CM 的 ID 号

```
typedef enum SysCMControllerTag
{
    CM1 = 0x00,
    CM2,
    CM3,
    CM4,
    NUM_OF_CM,
    INVALID_CM_ID
} CMController_t;
```

3. CM slot CM 槽号

```
typedef enum SysCMSlotTag
{
    SLOT3 = 0x03,
    SLOT4,
    SLOT5,
```

```
    SLOT6,
    SLOT7,
    SLOT8,
    SLOT9,
    SLOT10,
    SLOT11,
    SLOT12,
    SLOT13,
    SLOT14,
    SLOT15,
    NUM_OF_SLOT,
    INVALID_CM_SLOT
} CMSlot_t;
```

4. System time 系统时间

```
typedef struct SysSystemTimeTag
{
    uint32_t uiSecond;        /*time: s*/
    uint16_t usMiliSecond;    /*time: ms*/
} SysSystemTime_t;
```

5. Project version information 工程版本信息

```
typedef struct ProjVerInfoTag
{
    uint32_t uiProjID;          /* Project ID          工程ID */
    uint32_t uiMainVer;         /* Main Version    主版本  */
    uint32_t uiMinorVer;        /* Minor Version  次版本  */
    uint32_t uiProjCRC;         /* Project CRC       工程CRC */
}ProjVerInfo_t;
```

6. PM state PM 状态

```
typedef struct SysPMStateTag
{
    /* PM state */
    uint16_t usPMState[NUM_OF_PM];
    /* PM's system time */
    SysSystemTime_t stPMtime[NUM_OF_PM];
} SysPMState_t;
```

7. CM state PM 状态

```
typedef struct SysCMStateTag
{
```

```
    /* CM state */
    uint16_t usCMState[NUM_OF_CM];
    /* internal communication state */
    uint16_t usInterCommState[NUM_OF_CM][NUM_OF_PM];
    /* external communication state  各通讯口的外部通讯状态  */
    uint16_t usExterCommState[NUM_OF_CM][NUM_OF_NETWORK_PORT];
    /* CM's system time CM设置的系统时间  */
    SysSystemTime_t stCMtime[NUM_OF_CM];
} SysCMState_t;
```

8. System state 系统状态

```
typedef struct SysStateTag
{
    /* PMA~C状态  */
    SysPMState_t stPMState;
    /* CM1~4状态  */
    SysCMState_t stCMState;
    /* CM logic ID */
    uint32_t uiCMLogicID;
    /* Project info */
    PrjInfo_t stPrjInfo;
    PrjInfo_t stDownloadPrjInfo;
    TaskInformation_t stTaskInfo;
} SysState_t;
```

9. CM config table CM 配置表

```
typedef struct SysCMConfigTableTag
{
    /* the CM is configured or not  是否配置了该CM*/
    bool_t bValid[NUM_OF_CM];
    /* logic id  逻辑号  */
    uint8_t ucLogicID[NUM_OF_CM];
    /* slot id  槽号  */
    uint8_t ucSlotID[NUM_OF_CM];
} SysCMConfigTable_t;
```

10. CM to PM information CM 传输到 PM 的信息

```
typedef struct SysCMVernTag
{
    uint8_t ucRTSVern[4]; /*RTS version RTS版本信息  */
    uint8_t ucOSVern[4]; /*OS version OS版本信息  */
    uint16_t usFPGAModID; /*FPGA module ID FPGA模块ID */
```

```
    uint16_t usFPGAHardwareVern; /*FPGA hardware version FPGA硬件版本 */
    uint16_t usFPGAChipID; /*FPGA micr-chip ID FPGA芯片位号 */
    uint16_t usFPGAFWVern; /*FPGA firmware version FPGA固件版本 */
    uint16_t usMCUModID; /*MCU module ID MCU模块ID */
    uint16_t usMCUHardwareVern; /*MCU hardware version MCU硬件版本 */
    uint16_t usMCUChipID; /*MCU micr-chip ID MCU芯片位号 */
    uint16_t usMCUFWVern; /*MCU firmware version MCU固件版本 */
}SysCMVern_t;

typedef struct SysCM2PMTag
{
    SysCMVern_t stCMVern; /*CM version CM版本号 */
    uint16_t usDDRUsage; /*DDR usage 内存使用率 */
    uint16_t usCPUUsage; /*CPU usage CPU使用率 */
    uint16_t usSystemNet1Usage; /*NET1 usage 系统网1负荷 */
    uint16_t usSystemNet2Usage; /*NET2 usage 系统网2负荷 */
    uint8_t ucCMBUSAState; /*CMBUS-A state A系CMBUS状态 */
    uint8_t ucCMBUSBState; /*CMBUS-B state B系CMBUS状态 */
    uint8_t ucCMBUSCState; /* CMBUS-C state C系CMBUS状态 */
    uint16_t ucStateLED; /*LED state 状态灯状态 */
    uint16_t ucComLED; /*communication LED state 通讯灯状态 */
    uint16_t usPowerMonitor; /*Power monitor 电源监测 */
    float_t fTemperatureMonitor; /*temperature monitor 温度监测 */
    uint16_t usMCUMonitor; /*MCU monitor MCU监测 */
    uint16_t usPluseMonitor; /*Pluse monitor 脉冲监测 */
    uint8_t ucModbusState[6]; /*Modbus state Modbus状态 */
} SysCM2PM_t;
```

11. Project information 工程信息

```
typedef struct PrjInfoTag
{
    uint8_t ucIsConfigured;                 /*Project is configured or not 工程是否解析
完成*/
    uint8_t ucIsConfirmed;                  /*Project is confirmed or not 工程是否经用
户确认*/
    uint8_t ucIsIOConfigured;               /*IO is configured or not IO配置是否完成*/
    uint8_t ucRtDataSync;                   /*real time data is synchronized or not 实时
数据是否同步完成*/
    uint32_t uiPrjID;
    uint32_t uiMainVern;
    uint32_t uiMinorVern;
    uint32_t uiPrjCRC;
}PrjInfo_t;
```

12. Task information 任务信息

```
typedef struct TaskInformationTag
{
    uint8_t ucTaskNum;
    uint8_t ucTask1State;
    uint8_t ucTask2State;
}TaskInformation_t;
```

13. Task information 实时数据版本信息

```
typedef struct RTDataVerInfoTag
{
    uint32_t uiProjID;      /* Project ID  工程ID */
    uint32_t uiMainVer;     /* Main Version  主版本  */
    uint32_t uiMinorVer;    /* Minor Version  次版本  */
    uint32_t uiProjCRC;     /* Project CRC  工程CRC */
}RTDataVerInfo_t;
```

14. External communication network configuration information 外部通讯网络配置信息

```
typedef struct ExtCommNetConfigInfoTag
{
    ActiveFlag_t emActive;
    bool_t bUpdate;
    uint8_t ucReserve[3];
    uint32_t   uiIPAddr;
    uint32_t   uiSubnetMask;
    uint32_t   uiGateway;
}ExtCommNetConfigInfo_t;
```

15. OPC configuration information OPC 配置信息

```
typedef struct OPCNetConfigInfoTag
{
    ActiveFlag_t emActive;
    bool_t bUpdate;
    bool_t bPortWriteEnable;
    uint8_t ucReserve[2];
}OPCNetConfigInfo_t;
```

16. Modbus configuration information Modbus 配置信息

```
typedef struct ExtCommModbusTCPConfigInfoTag
{
```

```
    ActiveFlag_t emActive;
    bool_t bUpdate;
    uint8_t ucProtocol;
    uint8_t ucReserve[2];
    uint32_t    uiIPAddr;
}ExtCommModbusTCPConfigInfo_t;
```

17. Modbus and Modbus TCP message information Modbus 和 Modbus TCP 消息

```
/* message 消息*/
typedef struct SysModbusMsgTag
{
    uint16_t usLen;                                    /* message length 消息内容长度 */
    uint8_t ucContent[MAX_MODBUS_MSG_LEN];          /* message content 存放消息内容 */
} SysModbusMsg_t;

/* message queue 消息队列*/
typedef struct SysModbusMsgQueueTag
{
    uint16_t usHead;                                   /* header 队列头部*/
    uint16_t usTail;                                  /* tail 队列尾部 */
    uint16_t usNum;                                    /* element num 队列中元素个数 */
    SysModbusMsg_t stMsg[MAX_MODBUS_MSG_NUM];     /* message 消息 */
} SysModbusMsgQueue_t;

/* message 消息*/
typedef struct SysModbusTCPMsgTag
{
    uint16_t usLen;                                    /* message length 消息内容长度 */
    uint8_t ucContent[MAX_MODBUS_TCP_MSG_LEN];   /* message conten 存放消息内容 */
} SysModbusTCPMsg_t;

/*message queue 消息队列*/
typedef struct SysModbusTCPMsgQueueTag
{
    uint16_t usHead;                                   /* Header 队列头部*/
    uint16_t usTail;                                  /* tail 队列尾部 */
    uint16_t usNum;                                    /* element number 队列中元素个数 */
    SysModbusTCPMsg_t stMsg[MAX_MODBUS_TCP_MSG_NUM];     /*message 消息 */
} SysModbusTCPMsgQueue_t;
```

18. P2P message information P2P 消息

```
/* P2P message*/
typedef struct SysP2PMsgTag
{
```

```
    uint16_t usLen;                                 /* message length 消息内容长度 */
    uint8_t ucContent[MAX_P2P_MSG_LEN];             /*message content 存放消息内容 */
} SysP2PMsg_t;

/* P2P message box 信箱 */
typedef struct SysP2PMsgBoxTag
{
    bool_t bLock;                                   /* locked or not 信箱是否可以被访问 */
    bool_t bMsgBoxStatus;                           /* box state: full or empty 信箱的两种状
态：空或满 */
    SysP2PMsg_t stMsg;                              /* P2P message 消息 */
} SysP2PMsgBox_t;

/* message queue 消息队列*/
typedef struct SysP2PMsgQueueTag
{
    uint16_t usHead;                                /*Header 队列头部*/
    uint16_t usTail;                                /*Tail 队列尾部 */
    uint16_t usNum;                                 /*Element number 队列中元素个数 */
    SysP2PMsg_t stMsg[MAX_P2P_MSG_NUM];             /*message 消息 */
} SysP2PMsgQueue_t;

/* message type(request or ACK) 消息类型（请求或应答）*/
typedef enum SysP2PMsgTypeTag
{
    P2P_PM_REQ = 0,                                 /* PM request PM 发请求 */
    P2P_PM_RESP = 1,                                /* PM response PM 收应答 */
    P2P_EXT_REQ = 2,                                /*External request 收来自外部请求 */
    P2P_EXT_RESP = 3,                               /*External response 发应答给外部 */
    NUM_OF_P2PMSGTYPE
} SysP2PMsgType_t;
```

19. PC software related data structure 上位机软件相关的数据结构

1) Internal communication related data structure 内部通讯相关数据结构

```
typedef struct SysMsgTag
{
    uint16_t usLen;                                 /* 消息内容长度 */
    uint8_t ucContent[MAX_MSG_LEN];                 /* 存放消息内容 */
} SysMsg_t;

/*message box 信箱 */
typedef struct SysMsgBoxTag
{
```

```
    bool_t bLock;                                   /* locked or not 信箱是否可以被访问 */
    bool_t bMsgBoxStatus;                           /* box state: empty or full 信箱的两种状
态：空或满 */
    SysMsg_t stMsg;                                 /* message 消息 */
} SysMsgBox_t;

/* message type (request or response) 消息类型（请求或应答）*/
typedef enum SysMsgTypeTag
{
    SOE_REQ = 0,                                    /*request 请求 */
    SOE_RESP = 1,                                   /*response 应答 */
    CS1131_REQ = 2,                                 /* request 请求 */
    CS1131_RESP = 3,                                /* response 应答 */
    OPC_REQ = 4,                                    /* request 请求 */
    OPC_RESP = 5,                                   /* response 应答 */
    CLIENT_REQ = 6,                                 /* request 请求 */
    CLIENT_RESP = 7,                                /* response 应答 */
    AMS_REQ = 8,                                    /* request 请求 */
    AMS_RESP = 9,                                   /* response 应答 */
    INTER_CMD_REQ = 10,                             /* request 请求 */
    INTER_CMD_RESP = 11,                            /* response 应答 */
    NUM_OF_MSGTYPE
} SysMsgType_t;
```

2) External communication related data structure 外部通讯相关数据结构

```
/* Message 消息*/
typedef struct SysAppMsgTag
{
    uint16_t usLen;                                 /* message length 消息内容长度 */
    uint8_t ucContent[MAX_MSG_LEN];                 /*message content 存放消息内容 */
} SysAppMsg_t;

/*message box 信箱 */
typedef struct SysAppMsgBoxTag
{
    bool_t bLock;                                   /* locked or not MsgBox 是否可以被访问
*/
    bool_t bMsgBoxStatus;                           /* state: empty or full MsgBox 的两种状
态：空或满 */
    SysMsg_t stMsg;                                 /*message 消息 */
} SysAppMsgBox_t;

/* PC software type 上位机软件类型*/
typedef enum SysSoftwareTypeTag
{
```

```
    APP_SOE = 0,                                    /* SOE software */
    APP_CS1131 = 1,                                 /* CS1131 software */
    APP_OPC = 2,                                    /* OPC software */
    APP_CLIENT = 3,                                 /* client software */
    APP_AMS = 4,                                    /* AMS software */
    NUM_OF_SOFTWARETYPE
} SysSoftwareType_t;

/* message type (request or response)上位机软件的消息类型（请求或应答）*/
typedef enum SysAppMsgTypeTag
{
    APP_MSG_REQ = 0,                                /*request 请求 */
    APP_MSG_RESP = 1,                               /*response 应答 */
    NUM_OF_APPMSGTYPE
} SysAppMsgType_t;
```

20. Real time data 实时数据

```
/* real time data area 实时数据区 */
typedef struct RTDataAreaTag
{
    bool_t bLock;                   /*locked or not 是否可以被访问:true-不允许访问 false-允许访问 */
    bool_t bDataAreaStatus;         /*area state：read only or write only 两种状态：只读或只写: true-只读 false-只写*/
    uint8_t ucContent[MAX_RTDATA_G_AREA_LEN];       /*G content G 区数据内容 */
    uint8_t ucInputContent[MAX_RTDATA_I_AREA_LEN];  /*I content I 区数据内容 */
    uint8_t ucOutputContent[MAX_RTDATA_Q_AREA_LEN]; /*Q content Q 区数据内容 */
    uint8_t ucAppendInfo[MAX_RTDATA_APPEND_INFO_SIZE]; /*append area information 附加区内容 */
} RTDataArea_t;

typedef enum RTDataAreaIndexTag
{
    RTDATA_AREA_A = 0,                              /*real time data area A 实时数据区 A*/
    RTDATA_AREA_B = 1,                              /* real time data area A 实时数据区 B */
    NUM_OF_RTDATA_AREA,                             /* real time data area number 实时数据区个数 */
    INVALID_RTDATA_AREA_INDEX                       /*invalid index 实时数据区错误的索引号 */
} RTDataAreaIndex_t;
```

21. Configuration file 配置文件

```
typedef struct SysCfgFileStateTag
{
    bool_t bReadable;                          /* is it readable */
    bool_t bWriteable;                         /* is it writeable */
    bool_t bFileStatus;                        /* file state: write finished or not 配
置文件的两种状态：写完或未写完*/
    uint32_t uiLen;                            /* file length 配置文件长度 */
    uint32_t uiWrOffset;                       /* write offset 写偏移地址 */
} SysCfgFileState_t;

/* CM configuration file CM 配置文件*/
typedef struct SysCfgFileCMTag
{
    bool_t bReadable;                          /*readable flag 配置文件是否可
读 */
    bool_t bWriteable;                         /*write flag 配置文件是否可写
*/
    bool_t bFileStatus;                        /*file state: write finished or not 配
置文件的两种状态：写完或未写完*/
    uint32_t uiLen;                            /*file length 配置文件长度 */
    uint32_t uiWrOffset;                       /*write offset 写偏移地址 */
    uint8_t ucContent[MAX_CFG_CM_LEN];         /*file content 配置文件内容
*/
} SysCfgFileCM_t;

/*configuration file data base 配置文件数据库 */
typedef struct SysCfgDataBaseTag
{
    bool_t bReadable;                          /*DB readable flag 数据库是否
可读 */
    bool_t bWriteable;                         /*DB writeable flag 数据库是否
可写 */
    bool_t bDBStatus;                          /*DB state: empty or full 数据库
状态： 空或满*/
    SysCfgFileCtrlStation_t stCfgCtrlStation;  /*control station configuration file 控制
站配置文件 */
    SysCfgFilePRG_t stCfgPrg;                  /*PRG configuration file PRG 配
置文件 */
    SysCfgFileIOMod_t stCfgIOMod;              /*IO module configuration file
IO 模块配置文件 */
    SysCfgFileSOE_t stCfgSOE;                  /*SOE configuration file SOE 配
置文件 */
    SysCfgFileP2P_t stCfgP2P;                  /*P2P configuration file P2P 配置
```

```
文件 */
    SysCfgFileModbusMaster_t stCfgModbusMaster;          /*Modbus master configuration file Modbus master 配置文件 */
    SysCfgFileModbusSlave_t stCfgModbusSlave;            /*Modbus slave configuration file Modbus slave 配置文件 */
    SysCfgFileVote_t stCfgVote;                          /*Vote configuration file Vote 配置文件 */
    SysCfgFileRetain_t stCfgRetain;                      /*Retain configuration file Retain 配置文件 */
    SysCfgFileOPC_t stCfgOPC;                            /*OPC configuration file OPC 配置文件 */
    SysCfgFileCM_t stCfgCM;                              /*CM configuration file CM 配置文件 */
} SysCfgDB_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 3-2 Sub function list

表 3-2 子功能列表

Sub function No. 子功能编号	Function description 功能描述
SWDD-CM-SR_NSafR_NSecR_A_001	Module initialization. 模块初始化
SWDD-CM-SR_NSafR_NSecR_A_002	Common resource operation interface 通用资源操作接口
SWDD-CM-SR_NSafR_NSecR_A_003	Configuration information operation interface 配置信息资源操作接口
SWDD-CM-SR_NSafR_NSecR_A_004	Modbus information operation interface Modbus 信息资源操作接口
SWDD-CM-SR_NSafR_NSecR_A_005	P2P information operation interface P2P 信息资源操作接口
SWDD-CM-SR_NSafR_NSecR_A_006	PC software information operation interface 上位机软件信息资源操作接口
SWDD-CM-SR_NSafR_NSecR_A_007	Real time data operation interface 实时数据操作接口

SWDD-CM-SR_NSafR_NSecR_A_008	Configuration file operation interface 配置文件资源操作接口
SWDD-CM-SR_NSafR_NSecR_A_009	FPGA register operation interface FPGA 寄存器操作接口
SWDD-CM-SR_NSafR_NSecR_A_010	Common library 公共库函数
SWDD-CM-SR_NSafR_NSecR_A_011	LED 状态灯
SWDD-CM-SR_NSafR_SecR_A_012	Log 日志
SWDD-CM-SR_NSafR_NSecR_A_013	Version 版本号

4 Design of sub-function 子功能设计

4.1 Module initialization 模块初始化

SWDD-CM-SR_NSafR_NSecR_A_001

4.1.1 SysInit

4.1.1.1 Function Description 功能描述

System resource initialization.

系统资源管理模块初始化。

4.1.1.2 Argument Description 参数说明

- Definition 函数定义

int SysInit (void)

- Input argument 输入参数

No.

无。

- Output argument 输出函数

No.

无。

4.1.1.3 Processing flow 处理流程

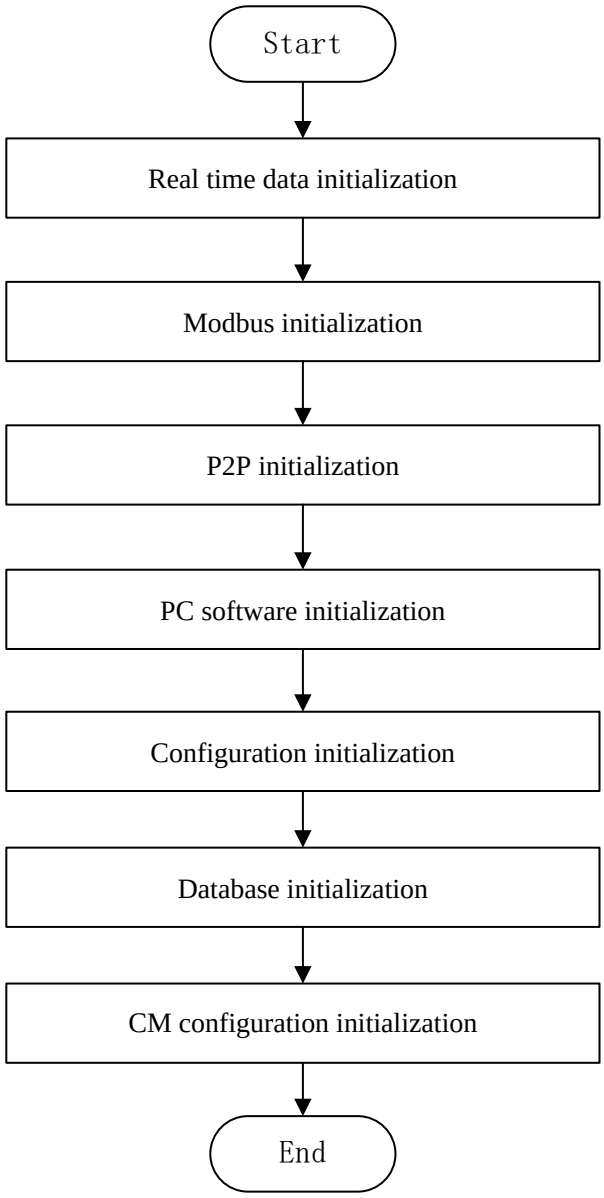

Figure4-1 System resource initialization processing flow
图 4-1 系统资源初始化流程图

4.2 Common resource operation interface 通用资源操作接口

SWDD-CM-SR_NSafR_NSecR_A_002

This sub-function is used for other modules to access the system resource.

本子功能用于其它模块对系统资源的读写访问。

4.2.1 SysGetPMState

4.2.1.1 Function Description 功能描述

Get PM's state.

获取 PM 的状态。

4.2.1.2 Argument Description 参数说明

- Definition 函数定义

bool_t SysGetPMState(PMController_t emPMID, uint16_t *pusState)

- Input argument 输入参数

PMController_t emPMID.

PM's ID

PM 的 ID 号

- Output argument 输出函数

uint16_t *pusState

point to PM's state

指向 PM 的状态

4.2.1.3 Processing flow 处理流程

This function is simple and the processing flow is omitted.

此函数逻辑简单，流程图省略。

4.2.2 SysSetPMState

4.2.2.1 Function Description 功能描述

Set PM's state.

设置 PM 的状态。

4.2.2.2 Argument Description 参数说明

- Definition 函数定义

bool_t SysSetPMState(PMController_t emPMID, uint16_t usPMState)

- Input argument 输入参数

PMController_t emPMID.

PM's ID

PM 的 ID 号

uint16_t usPMState

PM's state

PM 的状态

- Output argument 输出函数

No

无

4.2.2.3 Processing flow 处理流程

This function is simple and the processing flow is omitted.

此函数逻辑简单，流程图省略。

4.2.3 SysGetCMState

4.2.3.1 Function Description 功能描述

Get CM's state.

获取 CM 的状态。

4.2.3.2 Argument Description 参数说明

- Definition 函数定义

bool_t SysGetCMState(CMController_t emCMID, uint16_t *pusState)

- Input argument 输入参数

CMController_t emCMID.

CM's ID

CM 的 ID 号

- Output argument 输出函数

uint16_t *pusState

point to CM's state

指向 CM 的状态

4.2.3.3 Processing flow 处理流程

This function is simple and the processing flow is omitted.

此函数逻辑简单，流程图省略。

4.2.4 SysSetCMState

4.2.4.1 Function Description 功能描述

Set CM's state.

设置 CM 的状态。

4.2.4.2 Argument Description 参数说明

- Definition 函数定义

bool_t SysSetCMState(CMController_t emCMID, uint16_t usCMState)

- Input argument 输入参数

CMController_t emCMID.

CM's ID

CM 的 ID 号

uint16_t usCMState

CM's state

CM 的状态

- Output argument 输出函数

No

无

4.2.4.3 Processing flow 处理流程

This function is simple and the processing flow is omitted.

此函数逻辑简单，流程图省略。

4.3 Configuration information operation interface 配置信息操作接口

SWDD-CM-SR_NSafR_NSecR_A_003

This sub-function is used for other modules to access the system resource.

本子功能用于其它模块对系统资源的读写访问。

4.3.1 SysGetNetworkConfigInfo

4.3.1.1 Function Description 功能描述

Get network configuration information.

获取网络配置信息。

4.3.1.2 Argument Description 参数说明

- Definition 函数定义

bool_t SysGetNetworkConfigInfo(NetworkPortType_t emType, NetConfigInfo_t *pstNetworkParam);

- Input argument 输入参数

NetworkPortType_t emType.

The type of network port

网口类型（NET1/NET2）

- Output argument 输出函数

NetConfigInfo_t *pstNetworkParam

Network configuration parameter

网络配置参数

4.3.1.3 Processing flow 处理流程

This function is simple and the processing flow is omitted.

此函数逻辑简单，流程图省略。

4.3.2 SysGetModbusTcpConfigInfo

4.3.2.1 Function Description 功能描述

Get Modbus TCP configuration information.

获取 Modbus TCP 配置信息。

4.3.2.2 Argument Description 参数说明

- Definition 函数定义

bool_t SysGetModbusTcpConfigInfo(NetworkPortType_t emType, ModbusTCPConfigInfo_t *pstModbusTcpConfigInfo);

- Input argument 输入参数

NetworkPortType_t emType.

The type of network port

网口类型（NET1/NET2）

- Output argument 输出函数

ModbusTCPConfigInfo_t *pstModbusTcpConfigInfo

Modbus TCP configuration information

ModbusTCP 配置信息

4.3.2.3 Processing flow 处理流程

This function is simple and the processing flow is omitted.

此函数逻辑简单，流程图省略。

4.3.3 SysGetNTPTCPConfigInfo

4.3.3.1 Function Description 功能描述

Get NTP/TCP configuration information.

获取 NTP/TCP 配置信息。

4.3.3.2 Argument Description 参数说明

- Definition 函数定义

bool_t SysGetNTPTCPConfigInfo(NTPTCPConfigInfo_t *pstNTPTCPConfigInfo);

- Input argument 输入参数

No.

无

- Output argument 输出函数

NTPTCPConfigInfo_t *pstNTPTCPConfigInfo

NTP/TCP configuration information

NTP/TCP 配置信息

4.3.3.3 Processing flow 处理流程

This function is simple and the processing flow is omitted.

此函数逻辑简单，流程图省略。

4.4 Modbus information operation interface Modbus 信息操作接口

SWDD-CM-SR_NSafR_NSecR_A_004

This sub-function is used for other modules to access the system resource.

本子功能用于其它模块对 Modbus 系统资源的读写访问。

4.4.1 SysGetModbusMsgQueue

4.4.1.1 Function Description 功能描述

Get Modbus message from Modbus message queue.

从消息队列中获取 Modbus 信息。

4.4.1.2 Argument Description 参数说明

- Definition 函数定义

bool_t SysGetModbusMsgQueue (PMController_t emPMID, puint8_t pucContent, puint16_t pusLen);

- Input argument 输入参数

PMController_t emPMID.

PM's ID

PM 的 ID 号

- Output argument 输出函数

puint8_t pucContent

point to the message content

指向消息内容

puint16_t pusLen

point to message length

指向消息长度

4.4.1.3 Processing flow 处理流程

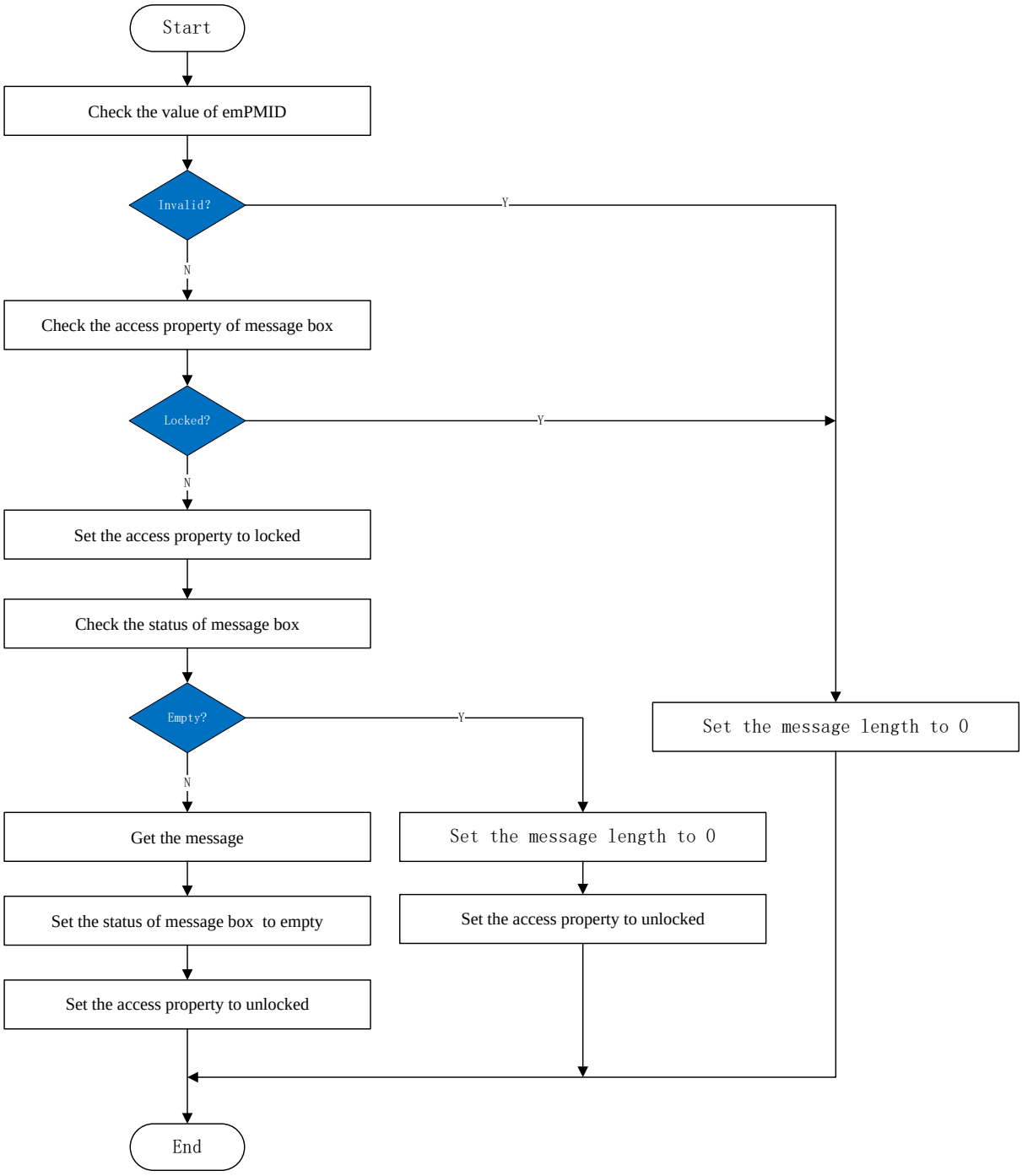

Figure4-2 Get Modbus message

图 4-2 获取 Modbus 消息

4.4.2 SysSetModbusMsgQueue

4.4.2.1 Function Description 功能描述

Set Modbus message to Modbus message queue.

向消息队列中写入 Modbus 信息。

4.4.2.2 Argument Description 参数说明

- Definition 函数定义

bool_t SysSetModbusMsgQueue(PMController_t emPMID, puint8_t pucContent, uint16_t usLen);

- Input argument 输入参数

PMController_t emPMID.

PM's ID

PM 的 ID 号

puint8_t pucContent

point to the message content

指向消息内容

uint16_t usLen

message length

消息长度

- Output argument 输出函数

No.

无

4.4.2.3 Processing flow 处理流程

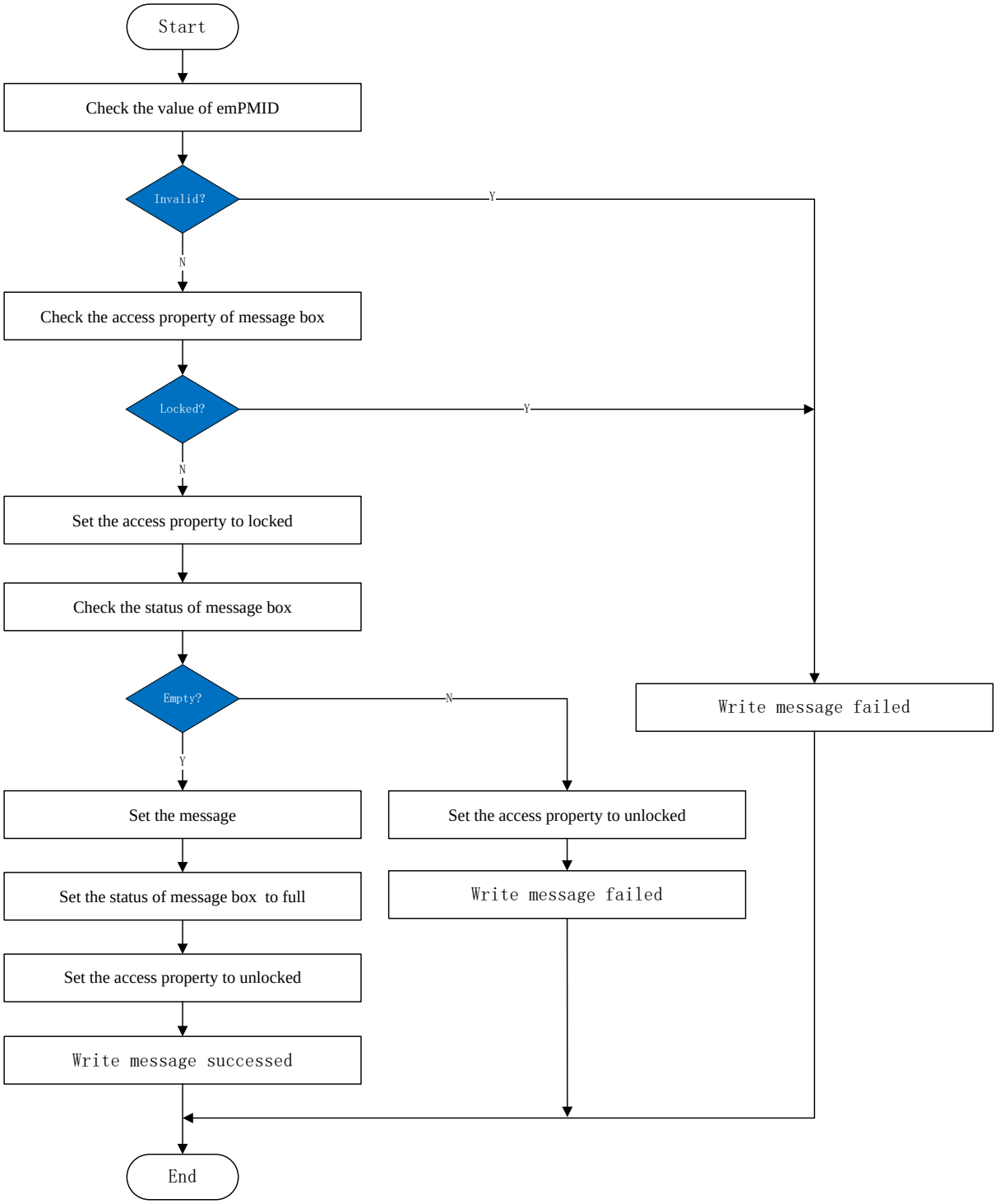

Figure4-3 Set Modbus message
图 4-3 写入 Modbus 消息

4.5 P2P information operation interface P2P 信息操作接口

SWDD-CM-SR_NSafR_NSecR_A_005

This sub-function is used for other modules to access the system resource.

本子功能用于其它模块对 P2P 系统资源的读写访问。

4.5.1 SysGetP2PMsg

4.5.1.1 Function Description 功能描述

Get P2P message from message queue.

从消息队列中获取 P2P 信息。

4.5.1.2 Argument Description 参数说明

- Definition 函数定义

bool_t SysGetP2PMsg (PMController_t emPMID, SysP2PMsgType_t emMsgType, puint8_t pucContent, puint16_t pusLen);

- Input argument 输入参数

PMController_t emPMID.

PM's ID

PM 的 ID 号

SysP2PMsgType_t emMsgType

Message type

消息类型

- Output argument 输出函数

puint8_t pucContent

point to the message content

指向消息内容

puint16_t pusLen

point to message length

指向消息长度

4.5.1.3 Processing flow 处理流程

This function is similar to SysGetModbusMsgQueue() and the processing flow is omitted.

此函数逻辑与读取 Modbus 消息函数逻辑一致，流程图省略。

4.5.2 SysSetP2PMsgQueue

4.5.2.1 Function Description 功能描述

Set P2P message to message queue.

向消息队列中写入 P2P 信息。

4.5.2.2 Argument Description 参数说明

- Definition 函数定义

bool_t SysSetP2PMsg (PMController_t emPMID, SysP2PMsgType_t emMsgType , puint8_t pucContent, uint16_t usLen);

- Input argument 输入参数

PMController_t emPMID.

PM's ID

PM 的 ID 号

SysP2PMsgType_t emMsgType

Message type

消息类型

puint8_t pucContent

point to the message content

指向消息内容

uint16_t usLen

message length

消息长度

- Output argument 输出函数

No.

无

4.5.2.3 Processing flow 处理流程

This function is similar to SysSetModbusMsgQueue() and the processing flow is omitted.

此函数逻辑与写入 Modbus 消息函数逻辑一致，流程图省略。

4.6 PC software information operation interface 上位机软件信息操作接口

SWDD-CM-SR_NSafR_NSecR_A_006

This sub-function is used for other modules to access the system resource.

本子功能用于其它模块对上位机软件相关系统资源的读写访问。

4.6.1 SysGetMsg

4.6.1.1 Function Description 功能描述

Get message from message box.

从邮箱中获取信息。

4.6.1.2 Argument Description 参数说明

- Definition 函数定义

bool_t SysGetMsg (PMController_t emPMID, SysMsgType_t emMsgType, puint8_t pucContent, puint16_t pusLen);

- Input argument 输入参数

PMController_t emPMID.

PM's ID

PM 的 ID 号

SysMsgType_t emMsgType

Message type

消息类型

- Output argument 输出函数

puint8_t pucContent

point to the message content

指向消息内容

puint16_t pusLen

point to message length

指向消息长度

4.6.1.3 Processing flow 处理流程

This function is similar to SysGetModbusMsgQueue() and the processing flow is omitted.

此函数逻辑与读取 Modbus 消息函数逻辑一致，流程图省略。

4.6.2 SysSetMsg

4.6.2.1 Function Description 功能描述

Set message to message box.

向邮箱中写入信息。

4.6.2.2 Argument Description 参数说明

- Definition 函数定义

bool_t SysSetMsg (PMController_t emPMID, SysMsgType_t emMsgType , puint8_t pucContent, uint16_t usLen);

- Input argument 输入参数

PMController_t emPMID.

PM's ID

PM 的 ID 号

SysP2PMsgType_t emMsgType

Message type

消息类型

puint8_t pucContent

point to the message content

指向消息内容

uint16_t usLen

message length

消息长度

- Output argument 输出函数

No.

无

4.6.2.3 Processing flow 处理流程

This function is similar to SysSetModbusMsgQueue() and the processing flow is omitted.

此函数逻辑与写入 Modbus 消息函数逻辑一致，流程图省略。

4.7 Real time data operation interface 实时数据操作接口

SWDD-CM-SR_NSafR_NSecR_A_007

This sub-function is used for other modules to access the real time data.

本子功能用于其它模块对实时数据的读写访问。

4.7.1 SysGetRTDataMsg

4.7.1.1 Function Description 功能描述

Get real time data

获取实时数据。

4.7.1.2 Argument Description 参数说明

- Definition 函数定义

bool_t SysGetRTDataMsg (PMController_t emPMID, SysRTDataMsgType_t emMsgType, puint8_t pucContent, puint16_t pusLen);

- Input argument 输入参数

PMController_t emPMID.

PM's ID

PM 的 ID 号

SysRTDataMsgType_t emMsgType

Message type

消息类型

- Output argument 输出函数

puint8_t pucContent

point to the message content

指向消息内容

puint16_t pusLen

point to message length

指向消息长度

4.7.1.3 Processing flow 处理流程

This function is similar to SysGetModbusMsgQueue() and the processing flow is omitted.

此函数逻辑与读取 Modbus 消息函数逻辑一致，流程图省略。

4.7.2 SysSetRTDataMsg

4.7.2.1 Function Description 功能描述

Set real time data.

写入实时数据。

4.7.2.2 Argument Description 参数说明

- Definition 函数定义

bool_t SysSetRTDataMsg (PMController_t emPMID, SysRTDataMsgType_t emMsgType , puint8_t pucContent, uint16_t usLen);

- Input argument 输入参数

PMController_t emPMID.

PM's ID

PM 的 ID 号

SysRTDataMsgType_t emMsgType

Message type

消息类型

puint8_t pucContent

point to the message content

指向消息内容

uint16_t usLen

message length

消息长度

- Output argument 输出函数

No.

无

4.7.2.3 Processing flow 处理流程

This function is similar to SysSetModbusMsgQueue() and the processing flow is omitted.

此函数逻辑与写入 Modbus 消息函数逻辑一致，流程图省略。

4.8 Configuration file operation interface 配置文件操作接口

SWDD-CM-SR_NSafR_NSecR_A_008

This sub-function is used for other modules to access the configuration file.

本子功能用于其它模块对配置文件的读写访问。

4.8.1 SysGetCfgFileInfo

4.8.1.1 Function Description 功能描述

Get configuration file information

获取配置文件信息。

4.8.1.2 Argument Description 参数说明

- Definition 函数定义

uint16_t SysGetCfgFileInfo (SysCfgFileType_t emCfgFileType, uint32_t uiRdOffset, puint8_t pucContent, uint16_t usLen);

- Input argument 输入参数

SysCfgFileType_t emCfgFileType,

Configuration file type

配置文件类型

uint32_t uiRdOffset

read offset

读取偏移地址

uint16_t usLen

message length

消息长度

- Output argument 输出函数

puint8_t pucContent,

point to the message content

指向消息内容

4.8.1.3 Processing flow 处理流程

This function is simple and the processing flow is omitted.

此函数逻辑简单，流程图省略。

4.8.2 SysSetCfgFileInfo

4.8.2.1 Function Description 功能描述

Set configuration file information.

写入配置文件信息。

4.8.2.2 Argument Description 参数说明

- Definition 函数定义

uint16_t SysSetCfgFileInfo(SysCfgFileType_t emCfgFileType, puint8_t pucContent, uint16_t usLen);

- Input argument 输入参数

SysCfgFileType_t emCfgFileType

file type

文件类型

puint8_t pucContent

point to the content

指向内容

uint16_t usLen

content length

内容长度

- Output argument 输出函数

No.

无

4.8.2.3 Processing flow 处理流程

This function is simple and the processing flow is omitted.

此函数逻辑简单，流程图省略。

4.9 FPGA register operation interface FPGA 寄存器操作接口

SWDD-CM-SR_NSafR_NSecR_A_009

This sub-function is used for other modules to access FPGA registers.

本子功能用于其它模块对 FPGA 寄存器的读写访问。

4.9.1 GetCMSlotAddr

4.9.1.1 Function Description 功能描述

Get CM's slot address.

获取 CM 的槽号。

4.9.1.2 Argument Description 参数说明

- Definition 函数定义

uint32_t GetCMSlotAddr(int32_t iTimeout);

- Input argument 输入参数

int32_t iTimeout

timeout

超时时间

- Output argument 输出函数

No

无

4.9.1.3 Processing flow 处理流程

This function is simple and the processing flow is omitted.

此函数逻辑简单，流程图省略。

4.9.2 CMSend

4.9.2.1 Function Description 功能描述

CM send data.

CM 发送数据。

4.9.2.2 Argument Description 参数说明

- Definition 函数定义

void CMSend(uint32_t uiPMAddr, puint8_t pSendBuf, uint32_t uiLen, int32_t iTimeout);

- Input argument 输入参数

uint32_t uiPMAddr

PM's address

PM 的地址

puint8_t pSendBuf

point to the content which will be sent

指向待发送的内容

uint16_t usLen

content length

内容长度

int32_t iTimeout

timeout

超时时间

- Output argument 输出函数

No.

无

4.9.2.3 Processing flow 处理流程

This function is simple and the processing flow is omitted.

此函数逻辑简单，流程图省略。

4.9.3 CMReceive

4.9.3.1 Function Description 功能描述

CM receive data.

CM 接收数据。

4.9.3.2 Argument Description 参数说明

- Definition 函数定义

void CMReceive(uint32_t uiPMAddr, uint8_t uiBufNo, puint8_t pucBuf);

- Input argument 输入参数

uint32_t uiPMAddr

PM’s address

PM 的地址

uint8_t uiBufNo

receive buffer number

接收缓冲区号码

puint8_t pucBuf

point to the content which will be received

指向接收到的内容

- Output argument 输出函数

No.

无

4.9.3.3 Processing flow 处理流程

This function is simple and the processing flow is omitted.

此函数逻辑简单，流程图省略。

4.10 Common library 公共库函数

SWDD-CM-SR_NSafR_NSecR_A_010

This sub-function is used for other modules to access common library.

本子功能用于其它模块对公共库函数的调用。

4.10.1 SysCrc32Cal

4.10.1.1 Function Description 功能描述

It is used to calculate CRC32.

计算 CRC32。

4.10.1.2 Argument Description 参数说明

- Definition 函数定义

uint32_t SysCrc32Cal(uint32_t uiCrc, const puint8_t pucBuf, uint32_t uiCount);

- Input argument 输入参数

uint32_t uiCrc,

the initial value

计算 CRC 的初始值

const puint8_t pucBuf

point to the content which will be used to calculate CRC32

指向计算 CRC32 的内容

uint32_t uiCount

the length of the content

数据内容的长度

- Output argument 输出函数

No

无

4.10.1.3 Processing flow 处理流程

This function is simple and the processing flow is omitted.

此函数逻辑简单，流程图省略。

4.11 LED 状态灯

SWDD-CM-SR_NSafR_NSecR_A_011

This sub-function is used to control LED.

本子功能用于控制 LED 状态灯。

4.11.1 LEDManagerInit

4.11.1.1 Function Description 功能描述

LED initialization.

LED 初始化。

4.11.1.2 Argument Description 参数说明

- Definition 函数定义

void LEDManagerInit(void);

- Input argument 输入参数

No

无

- Output argument 输出函数

No

无

4.11.1.3 Processing flow 处理流程

This function is simple and the processing flow is omitted.

此函数逻辑简单，流程图省略。

4.11.2 LEDManagerCycle

4.11.2.1 Function Description 功能描述

Update the states of LED.

周期的更新 LED 状态。

4.11.2.2 Argument Description 参数说明

- Definition 函数定义

void LEDManagerCycle(void);;

- Input argument 输入参数

No

无

- Output argument 输出函数

No

无

4.11.2.3 Processing flow 处理流程

This function is simple and the processing flow is omitted.

此函数逻辑简单，流程图省略。

4.12 Log 日志

SWDD-CM-SR_NSafR_SecR_A_012

This sub-function is used to read/write log.

本子功能用于读写日志。

4.12.1 LogWrite

4.12.1.1 Function Description 功能描述

Write log.

写日志。

4.12.1.2 Argument Description 参数说明

- Definition 函数定义

int32_t LogWrite(uint32_t usLogID);

- Input argument 输入参数

uint32_t usLogID

Log ID

- Output argument 输出函数

Returns the write result. If an error is returned, the result shall be sent to the diagnostic software.

返回写结果。如果返回错误，将错误信息传给诊断软件。

4.12.1.3 Processing flow 处理流程

This function is simple and the processing flow is omitted.

此函数逻辑简单，流程图省略。

4.12.2 LogRead

4.12.2.1 Function Description 功能描述

Read log.

读日志。

4.12.2.2 Argument Description 参数说明

- Definition 函数定义

int32_t LogRead(LogInfo_t *pstLogInfo, uint16_t usLogicPos, uint32_t uiLogType, puint32_t puiNextIndex);

- Input argument 输入参数

uint16_t usLogicPos

Log’s position

日志位置

uint32_t uiLogType

Log’s type

日志类型

- Output argument 输出函数

LogInfo_t *pstLogInfo

Point to log information

指向日志信息

puint32_t puiNextIndex

Point to next log’s index

指向下一条日志的索引号

4.12.2.3 Processing flow 处理流程

This function is simple and the processing flow is omitted.

此函数逻辑简单，流程图省略。

4.13 Version 版本号

SWDD-CM-SR_NSafR_NSecR_A_013

This sub-function is used to show the version of CM.

本子功能用于显示 CM 的版本号。

4.13.1 ShowCMVersion

4.13.1.1 Function Description 功能描述

Show the version of CM.

显示 CM 的版本号。

4.13.1.2 Argument Description 参数说明

- Definition 函数定义

void ShowCMVersion (void);

- Input argument 输入参数

No

无

- Output argument 输出函数

No

无

4.13.1.3 Processing flow 处理流程

This function is simple and the processing flow is omitted.

此函数逻辑简单，流程图省略。

——以下无正文