Document Title: CM_FW internal communication module design description of Safety Control System

Document Number: 16-Q04-000119

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 内部通讯模块设计说明书

文件编号：16-Q04-000119

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 internal communication 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 内部通讯模块的设计方案。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 internal communication module (marked red) in the software hierarchy is shown below.

内部通讯模块（标红）在软件层次中的位置如下图所示。

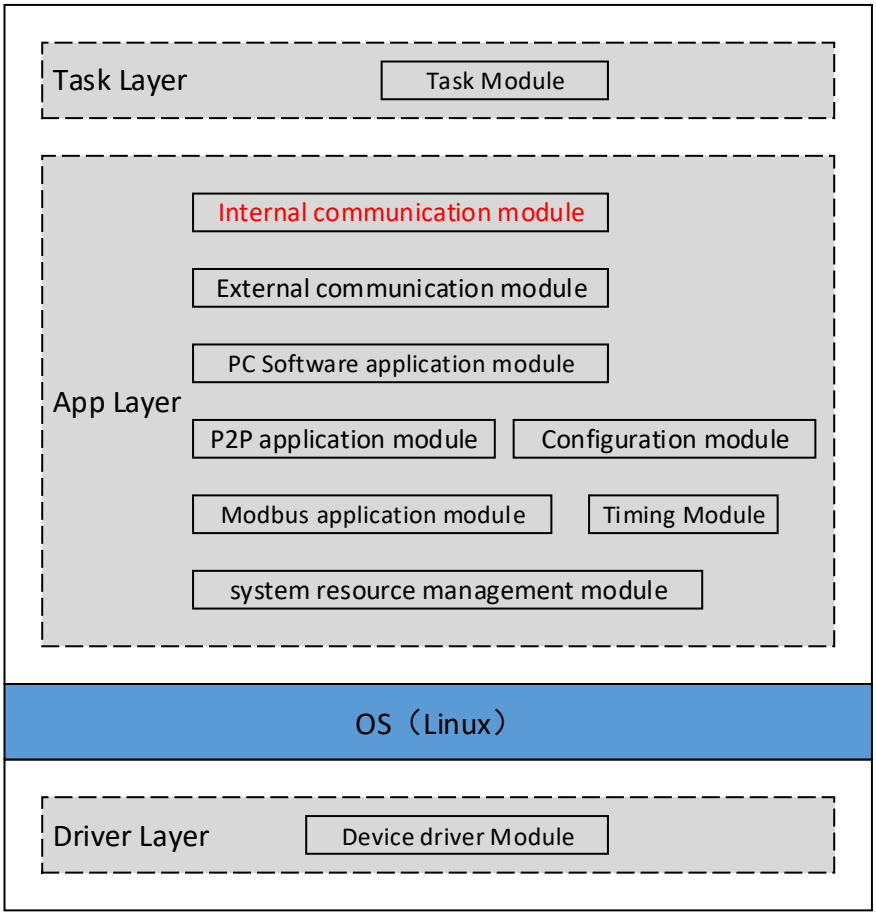

Figure 2-1 the location of the module

图 2-1 模块位置

3 Module design 模块设计

3.1 Function description 功能描述

CM communicate with PM via this module.

CM 与 PM 通过该模块进行内部通讯。

3.2 Design concept 设计思路

3.2.1 SOE software SOE软件

①The SOE application module set the request to request-buffer;

①SOE 应用模块将对各 PM 的 SOE 请求放入系统资源管理模块中的缓冲区；

②The internal communication module set the SOE request to send-buffer;

②内部通讯模块将 SOE 请求拷贝到发送缓冲区的相应位置；

③Move the data in send-buffer to FPGA buffer;

③将发送缓冲区中的数据发送的 FPGA 中的对应缓冲区中；

④CM received the data from PM;

④CM 接收来自 PM 的数据；

⑤Parse the received data, set the received data to receive-buffer;

⑤对接收到的数据进行解析，将 SOE 应答数据放入系统资源管理模块的对应缓冲区中；

⑥The application module get the SOE response.

⑥SOE 应用模块获取 SOE 应答数据。

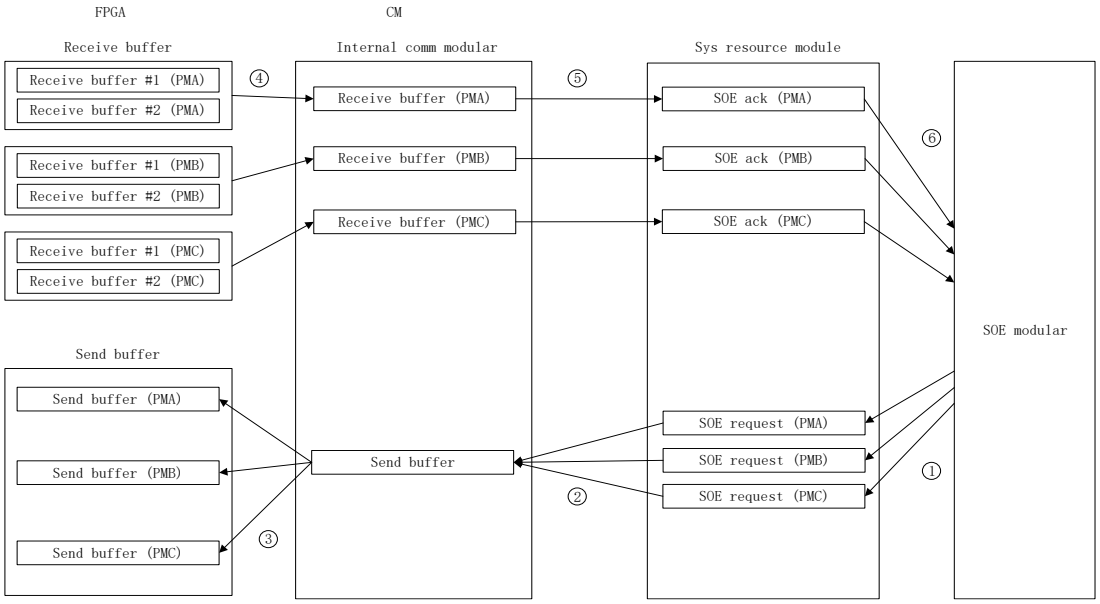

Figure 3-1 SOE software communication data flow

图 3- 1 与 SOE 软件通讯数据流

3.2.2 Peer-to-Peer communication P2P通讯

PM received P2P data via CM. The data flow please see the following figure:

PM 通过 CM 从外界接收数据如下图所示：

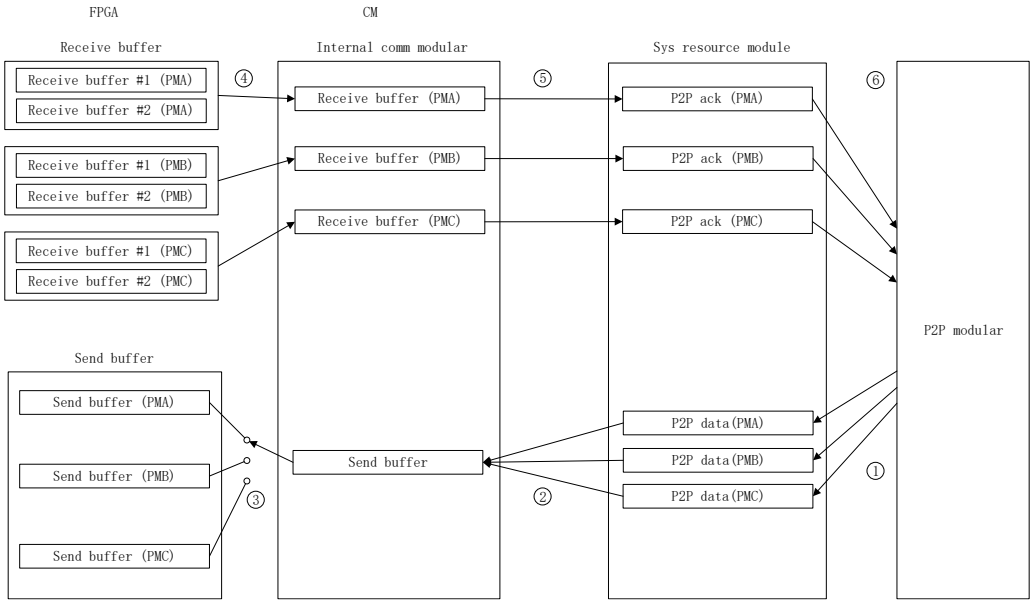

Figure 3-2 PM receive P2P data

图 3- 2 PM 接收 P2P 数据

3.2.3 CS1131 software 组态软件

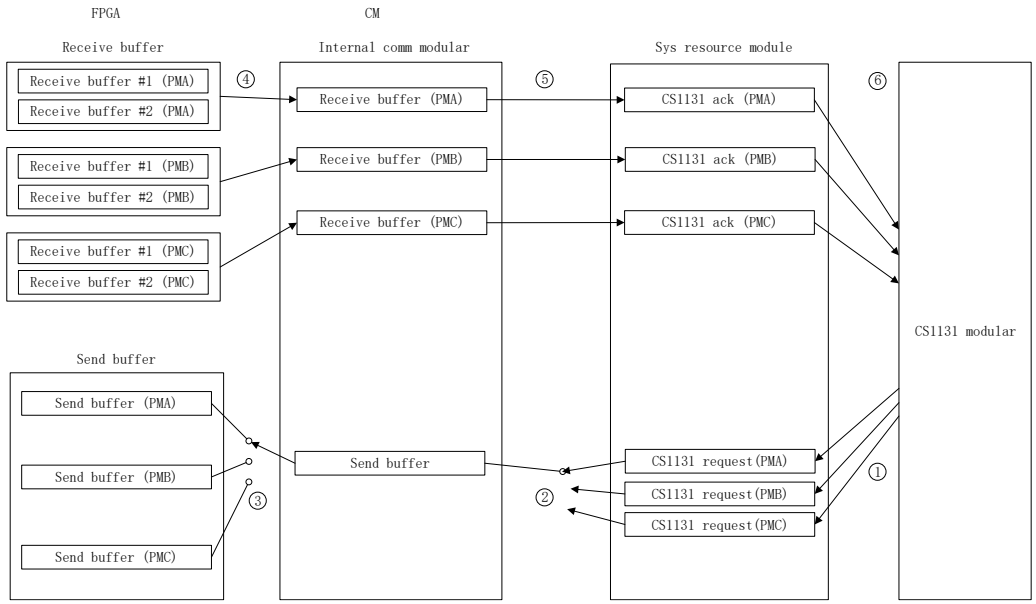

Figure 3-3 CM communicate with CS1131

图 3- 3 CM 与 CS1131 组态软件通讯

3.2.4 Request configure file 获取配置文件

①CM request the configuration file from PM

①CM 应用模块向相应的 PM 发送获取配置文件请求；

②Internal communication module send the request to PM;

②内部通讯模块将请求发送给 PM；

③Internal communication module get the response;

③内部通讯模块获取来自 PM 的应答；

④Internal communication module set the response (configuration file) to the related buffer;

④内部通讯模块将应答（配置文件）放入资源管理模块相应的缓冲区中。

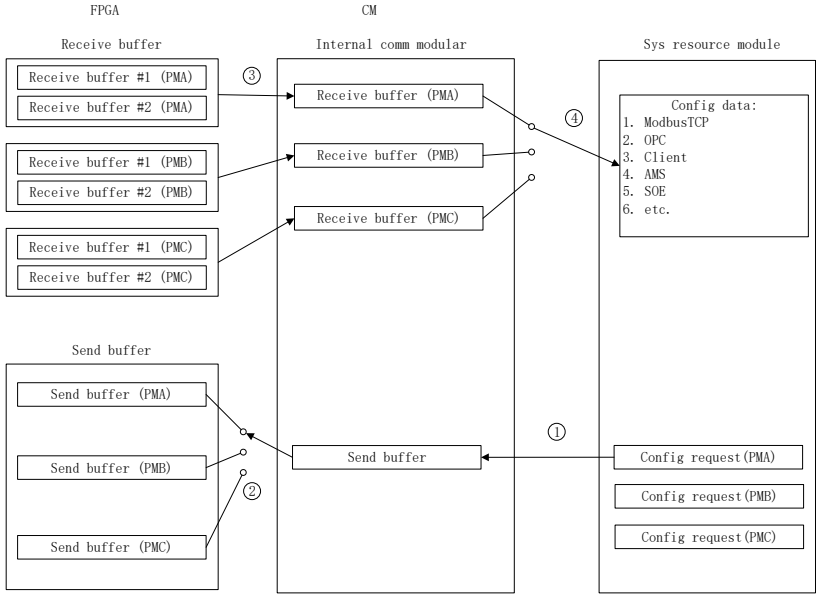

Figure 3-4 Configure file transfer data flow

图 3- 4 配置文件传输数据流

3.2.5 AMS software AMS软件

The internal communication module take the request from AMS software and send it to PM, it used the same strategy as CS1131 software.

处理 AMS 软件发送来的请求，与处理 CS1131 软件发送来的请求采用相同的处理策略。

3.2.6 OPC Server OPC SERVER

The internal communication module take the request from OPC Server and send it to PM, it used the same strategy as CS1131 software.

处理 OPC SERVER 发送来的请求，与处理 CS1131 软件发送来的请求采用相同的处理策略。

3.2.7 Diagnostic Software 诊断软件

The internal communication module take the request from Diagnostic Software and send it to PM, it used the same strategy as CS1131 software.

处理诊断软件发送来的请求，与处理 CS1131 软件发送来的请求采用相同的处理策略。

3.2.8 NTP/SNTP server (master) NTP/SNTP SERVER（主站）

The internal communication module take the request from NTP/SNTP server and send it to PM, it used the same strategy as CS1131 software.

处理 NTP/SNTP SERVER 发送来的请求，与处理 CS1131 软件发送来的请求采用相同的处理策略。

3.2.9 Modbus

The internal communication module take the request from Modbus software and send it to PM, it used the similar strategy as CS1131 software.

处理外部 Modbus 主站发送来的请求，与处理 CS1131 软件发送来的请求采用相同的处理策略。

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 InternalCommInit (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. void InternalCommCycle (void)

Input argument 输入参数	Output argument 输出参数	Description 描述
No. 无。	No. 无。	Internal communication interface function. 内部通讯周期运行函数

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.	RecvBuffer_t	s_stRecvBuffer1	Receive buffer 1. 接收缓冲区 1
2.	RecvBuffer_t	s_stRecvBuffer2	Receive buffer 2. 接收缓冲区 2
3.	SendBuffer_t	s_stSendBuffer	Send buffer 发送缓冲区
4.	uint16_t	s_usSendPackageSerialNum [NUM_OF_PM]	The serial number of send package 发送包序号
5.	uint16_t	s_usRecvPackageSerialNum	The serial number of receive package

		[NUM_OF_PM]	接收包序号
6.	uint16_t	s_usInterCommState [NUM_OF_PM] [NUM_OF_SLOT]	The internal communication state 内部通讯状态
7.	RTDataArea_t	*s_pstRTDataArea	Point to real time data area 指向实时数据区

3.5 Data structure 数据结构

```
#pragma pack(1)
/*communication buffer handling data structure 对缓冲区的动态操作过程*/
typedef struct CommHandlingTag
{
    uint16_t usUsed;                /*used 已使用空间 */
    uint16_t usRest;                /*rest 剩余空间 */
    uint16_t usOffset;              /*offset 可以使用的偏移 */
} CommHandling_t;

/*offset and data length CM 各类收发信息在缓冲区中的偏移地址及长度*/
typedef struct InfoStructTag
{
    uint16_t usLen;                 /*data length 数据长度 */
    uint16_t usOffset;              /*offset 在缓冲区中的偏移地址 */
} InfoStruct_t;

/*CM 接收数据格式*/
typedef struct ReceiveDataHeaderTag
{
    uint16_t usCMAddr;              /*CM address  CM 地址：1~4 */
    uint16_t usPackageSerialNum;    /*serial number 数据包序列号  */
    uint16_t usDataLen;             /*data length 数据内容长度 */
    InfoStruct_t stSysStateBlock;   /*system state 系统状态信息（PM 状态、CM 状态及校时信息）*/
    InfoStruct_t stDetectMsgBlock;  /*detection message PM 发送的探测消息 */
    InfoStruct_t stSOEResp;         /*SOE response SOE 应答 */
    InfoStruct_t stRealTimeData;    /*real time data 实时数据 */
    InfoStruct_t stCS1131Resp;      /*CS1131 response 组态软件应答 */
    InfoStruct_t stP2PReq;          /*P2P request P2P 请求 */
    InfoStruct_t stP2PResp;         /*P2P response P2P 应答 */
    InfoStruct_t stOPCResp;         /*OPC response OPC 应答 */
    InfoStruct_t stClientReq;       /*client request 客户端请求 */
    InfoStruct_t stClientResp;      /*client response 客户端应答 */
```

```
    InfoStruct_t stAMSResp;              /*AMS response AMS 应答 */
    InfoStruct_t stCfgTableBlock;     /*configure table 配置表 */
    InfoStruct_t stConfigRespBlock; /*configure file response 应答 CM 配置信息 */
    uint16_t usReserved[16];            /*reserved 保留 32B */
    uint32_t uiCRC32;                    /*CRC 32 位 CRC 校验码 */
} RecvHeader_t;

typedef struct ReceiveDataBodyTag
{
    uint8_t ucDataBuffer[CM_RECV_BUFF_BODY_SIZE];
} RecvBody_t;

typedef struct ReceiveBufferTag
{
    RecvHeader_t stRecvHeader;         /*receive data header 接收的头部信息 */
    RecvBody_t stRecvBody;              /*receive data body 接收的实际信息 */
} RecvBuffer_t;

/*send buffer struction 对发送缓冲区的动态填充操作过程*/
typedef struct SendDataHeaderTag
{
    uint16_t usCMAddr;                   /*CM address CM 地址：1~4 */
    uint16_t usPackageSerialNum;      /*serial number 数据包序列号 */
    uint16_t usDataLen;                  /*data length 数据内容长度 */
    InfoStruct_t stCMStateInfo;       /*CM state 系统状态信息（CM 状态及校时信息）*/
    InfoStruct_t stDetectMsgAckBlock;/*detection message ack block 对探测消息的应答 */
    InfoStruct_t stSOEReq;              /*SOE request SOE 请求 */
    InfoStruct_t stRealTimeDataAck; /*real time data ack 实时数据应答 */
    InfoStruct_t stCS1131Req;          /*CS1131 request 组态软件请求 */
    InfoStruct_t stP2PReq;              /*P2P request P2P 请求 */
    InfoStruct_t stP2PResp;             /*P2P response  P2P 应答 */
    InfoStruct_t stModbus;              /*Modbus data Modbus 数据 */
    InfoStruct_t stOPCReq;              /*OPC request OPC 请求 */
    InfoStruct_t stClientReq;          /*client request 客户端请求 */
    InfoStruct_t stClientResp;         /*client response 客户端应答 */
    InfoStruct_t stAMSReq;              /*AMS request AMS 请求 */
    InfoStruct_t stConfigReqBlock;   /*configure request block 请求 CM 配置信息 */
    InfoStruct_t stInterCmd;            /*internal command 内部命令信息 */
    uint16_t usReserved[14];            /*reserved 保留 28B */
    uint32_t uiCRC32;                    /*CRC  32 位 CRC 校验码 */
} SendHeader_t;

typedef struct SendDataBodyTag
{
    uint8_t ucDataBuffer[CM_SEND_BUFF_BODY_SIZE];
```

```
} SendBody_t;

typedef struct SendBufferTag
{
    SendHeader_t stSendHeader;          /*send data header 发送的头部信息 */
    SendBody_t stSendBody;               /*send data body 发送的实际内容信息 */
} SendBuffer_t;

typedef struct DetectMsgTag
{
    uint32_t uiFlag;                    /*detection flag 探测标识：    0x12345678-探测数据包
*/
} DetectMsg_t;

/*detection message block 探测消息数据块 */
typedef struct DetectMsgBlockTag
{
    /*start flag 起始标识 */
    uint16_t usStartFlag;
    /*data length 数据长度   sizeof(DetectMsg_t)*/
    uint16_t usLen;
    /*detection message */
    DetectMsg_t stDetectMsg;
    /*CRC 校验码 */
    uint32_t uiCrc32;
} DetectMsgBlock_t;

typedef struct DetectMsgAckTag
{
    uint32_t uiFlag;                    /*detection flag 探测标识：    0x87654321-探测数据包
应答 */
} DetectMsgAck_t;

/*detection ack block 探测消息应答数据块 */
typedef struct DetectMsgAckBlockTag
{
    /*start flag 起始标识 */
    uint16_t usStartFlag;
    /*data length 数据长度   sizeof(DetectMsgAck_t)*/
    uint16_t usLen;
    /*ack 应答 */
    DetectMsgAck_t stDetectMsgAck;
    /*CRC 校验码 */
    uint32_t uiCrc32;
} DetectMsgAckBlock_t;
```

```
/*configure command  获取配置信息命令  */
typedef struct ConfigReqTag
{
    uint32_t uiRequest;                 /* CONFIG_INFO_START
                                            CONFIG_INFO_CONTINUE
                                            CONFIG_INFO_RETRANSFER
                                            TRANSFER_CONFIG_INFO_FINISHED
                                        */
} ConfigReq_t;

/*request block  配置信息请求数据块  */
typedef struct ConfigReqBlockTag
{
    /*start flag  起始标识  */
    uint16_t usStartFlag;
    /*data length  数据长度    sizeof(ConfigReq_t)*/
    uint16_t usLen;
    /*request command  请求命令  */
    ConfigReq_t stConfigReq;
    /*CRC  校验码  */
    uint32_t uiCrc32;
} ConfigReqBlock_t;

/*configure file header  来自 PM 的配置信息头部  */
typedef struct CfgFileRespHeaderTag
{
    /*start flag  起始标识  */
    uint16_t usStartFlag;
    /*configure file type  配置文件类型  */
    uint16_t usCfgFileType;
    /*offset address  数据内容偏移地址  */
    uint32_t uiOffset;
    /*data length  数据长度  */
    uint16_t usLen;
    /*last package flag  最后一包标志  */
    uint16_t usLastPackFlag;
} CfgFileRespHeader_t;

/*configure table data block  配置表数据块  */
typedef struct CfgFileTableBlockTag
{
    /*start flag  起始标识  */
    uint16_t usStartFlag;
    /*data length  数据长度    sizeof(SysCfgFileTable_t)*/
```

```
    uint16_t usLen;
    /*configure table  配置表  */
    SysCfgFileTable_t stCfgTable;
    /*CRC  校验码  */
    uint32_t uiCrc32;
} CfgFileTableBlock_t;

#pragma pack()

#pragma pack(1)
/*real time data header information  每包实时数据头部信息  */
typedef struct RTDataHeaderTag
{
    uint16_t usStartFlag;                    /*start flag  开始标识  */
    uint16_t usServiceCode;                   /*service code  服务码*/
    uint32_t uiSerialNum;                     /*serial number  数据包序列号*/
    uint32_t  uiOffset;                       /*offset  address  在数据区中的偏移地
址  */
    uint16_t usTransferFlag;                 /*transfer flag  传输标识  */
    uint16_t usLen;                           /*data length  数据内容长度*/
    uint16_t usAreaType;                      /*data area type  数据区类型  */
} RTDataHeader_t;

/*response information  每包实时数据应答信息  */
typedef struct RTDataAckTag
{
    uint16_t usStartFlag;                    /*start flag  开始标识  */
    uint16_t usServiceCode;                   /*service code  服务码*/
    uint32_t uiSerialNum;                     /*serial number  数据包序列号*/
    uint16_t usLen;                           /*ack package length  应答包长度*/
    uint16_t usAck;                           /*ack  应答*/
    uint16_t usErrCode;                       /*error code  错误码*/
    uint32_t uiCRC32;                         /* CRC32*/
} RTDataAck_t;

#pragma pack()
```

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-IC_NSafR_SecR_A_001	Module initialization. 模块初始化
SWDD-CM-IC_NSafR_SecR_A_002	Internal communication cycle function. 内部通讯周期运行函数

4 Design of sub-function 子功能设计

4.1 Module initialization 模块初始化

SWDD-CM-IC_NSafR_SecR_A_001

4.1.1 InternalCommInit

4.1.1.1 Function Description 功能描述

Internal communication initialization function.

内部通讯初始化函数。

4.1.1.2 Argument Description 参数说明

- Definition 函数定义

void InternalCommInit (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

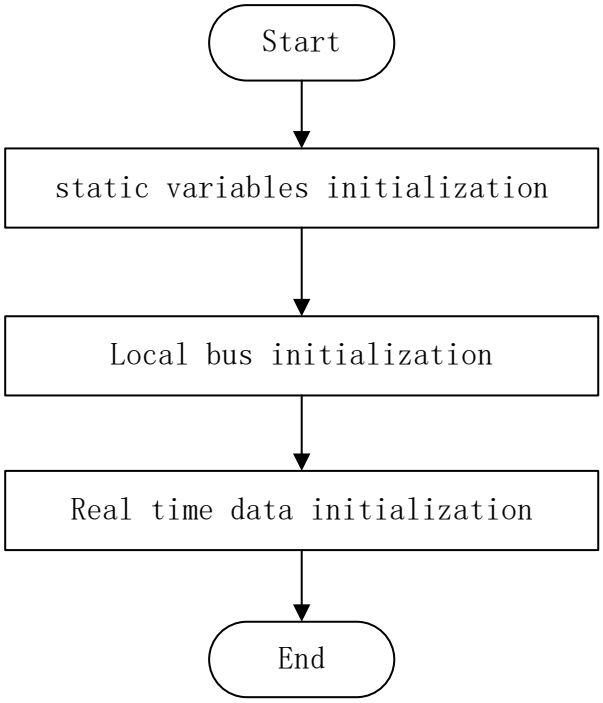

Figure4-1 internal communication initialization processing flow
图 4-1 内部通讯初始化流程图

4.1.2 InterCommRTDataInit

4.1.2.1 Function Description 功能描述

Real time data initialization.

实时数据初始化

4.1.2.2 Argument Description 参数说明

- Definition 函数定义

void InterCommRTDataInit (void)

- Input argument 输入参数

No.

无。

- Output argument 输出函数

No.

无。

4.1.2.3 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 operation 模块周期运行

SWDD-CM-IC_NSafR_SecR_A_002

This sub-function is used to process the internal data between CM and PM cyclically.

本子功能用于周期性处理 CM 和 PM 间的内部通讯数据。

4.2.1 InternalCommCycle

4.2.1.1 Function Description 功能描述

This function is used to process the request or response from SOE software, CS1131 software, AMS software, OPC Server, Diagnostic Software, NTP/SNTP Server, Modbus, P2P etc.

本函数用于周期处理来自 SOE 软件、CS1131 软件、AMS 软件、OPC SERVER、诊断软件、NTP/SNTP SERVER、Modbus、P2P 等的请求或应答。

4.2.1.2 Argument Description 参数说明

- Function Definition 函数定义

void InternalCommCycle (void);

- Input argument 输入参数

No.

无。

- Output argument 输出函数

No.

无。

4.2.1.3 Processing flow 处理流程

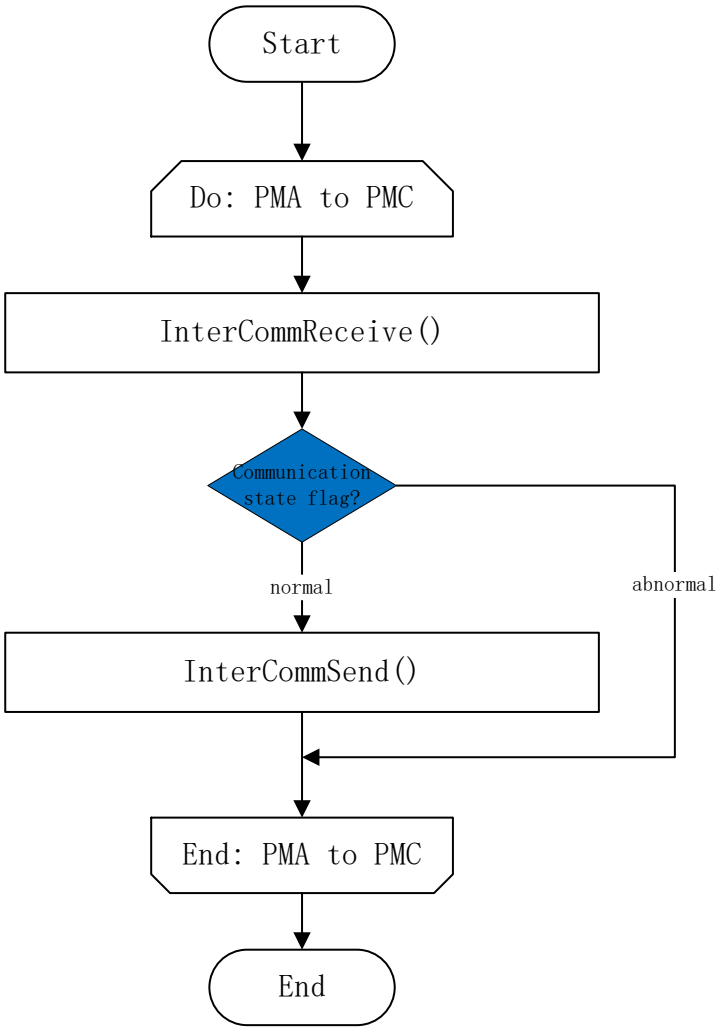

Figure4-2 internal communication cycle processing flow
图 4-2 内部通讯周期运行流程图

4.2.2 InterCommReceive

4.2.2.1 Function Description 功能描述

This function is used to receive the request or response from SOE software, CS1131 software, AMS software, OPC Server, Diagnostic Software, NTP/SNTP Server, Modbus, and P2P etc.

本函数用于周期接收来自 SOE 软件、CS1131 软件、AMS 软件、OPC SERVER、诊断软件、NTP/SNTP SERVER、Modbus、P2P 等的请求或应答。

4.2.2.2 Argument Description 参数说明

- Function Definition 函数定义

static bool_t InterCommReceive(PMController_t emPMID);

- Input argument 输入参数

PMController_t emPMID

PM's ID

PM 的 ID。

- Output argument 输出函数

No.

无。

4.2.2.3 Processing flow 处理流程

The processing flow is omitted.

流程图省略。

4.2.3 InterCommReceiveHandling

4.2.3.1 Function Description 功能描述

This function is used to process the received data from PM.

本函数用于周期处理来自 PM 的数据。

4.2.3.2 Argument Description 参数说明

- Function Definition 函数定义

static void InterCommReceiveHandling(RecvBuffer_t *pstRecvBuf, PMController_t emPMID);

- Input argument 输入参数

RecvBuffer_t *pstRecvBuf

Point to the receive buffer

指向接收缓冲区

PMController_t emPMID

PM's ID

PM 的 ID

- Output argument 输出函数

No.

无。

4.2.3.3 Processing flow 处理流程

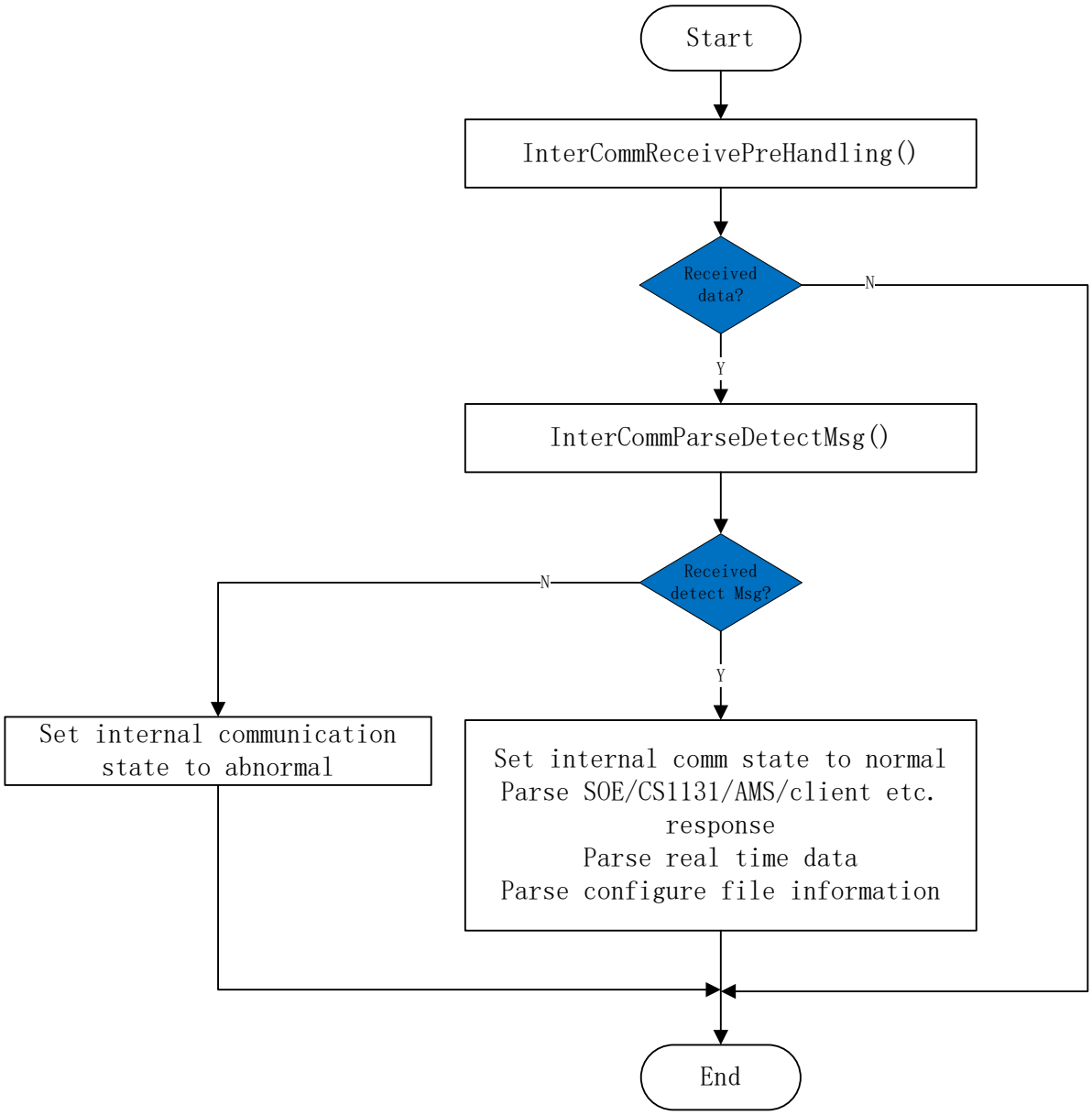

Figure4-3 received data processing flow

图 4-3 接收数据处理流程图

4.2.4 InterCommSend

4.2.4.1 Function Description 功能描述

This function is used to send the request or response from SOE software, CS1131 software, AMS software, OPC Server, Diagnostic Software, NTP/SNTP Server, Modbus, and P2P etc. to PM

本函数用于周期发送来自 SOE 软件、CS1131 软件、AMS 软件、OPC SERVER、诊断软件、NTP/SNTP SERVER、Modbus、P2P 等的请求或应答到 PM。

4.2.4.2 Argument Description 参数说明

- Function Definition 函数定义

static void InterCommSend(PMController_t emPMID);

- Input argument 输入参数

PMController_t emPMID

PM's ID

PM 的 ID。

- Output argument 输出函数

No.

无。

4.2.4.3 Processing flow 处理流程

The processing flow is omitted.

流程图省略。

4.2.5 InterCommSendHandling

4.2.5.1 Function Description 功能描述

This function is used to send the request/response data to PM.

本函数用于周期发送请求或应答数据到 PM。

4.2.5.2 Argument Description 参数说明

- Function Definition 函数定义

static void InterCommSendHandling(SendBuffer_t *pstSendBuf, PMController_t emPMID);

- Input argument 输入参数

SendBuffer_t *pstSendBuf

Point to send data buffer

指向发送缓冲区

PMController_t emPMID

PM's ID

PM 的 ID。

- Output argument 输出函数

No.

无。

4.2.5.3 Processing flow 处理流程

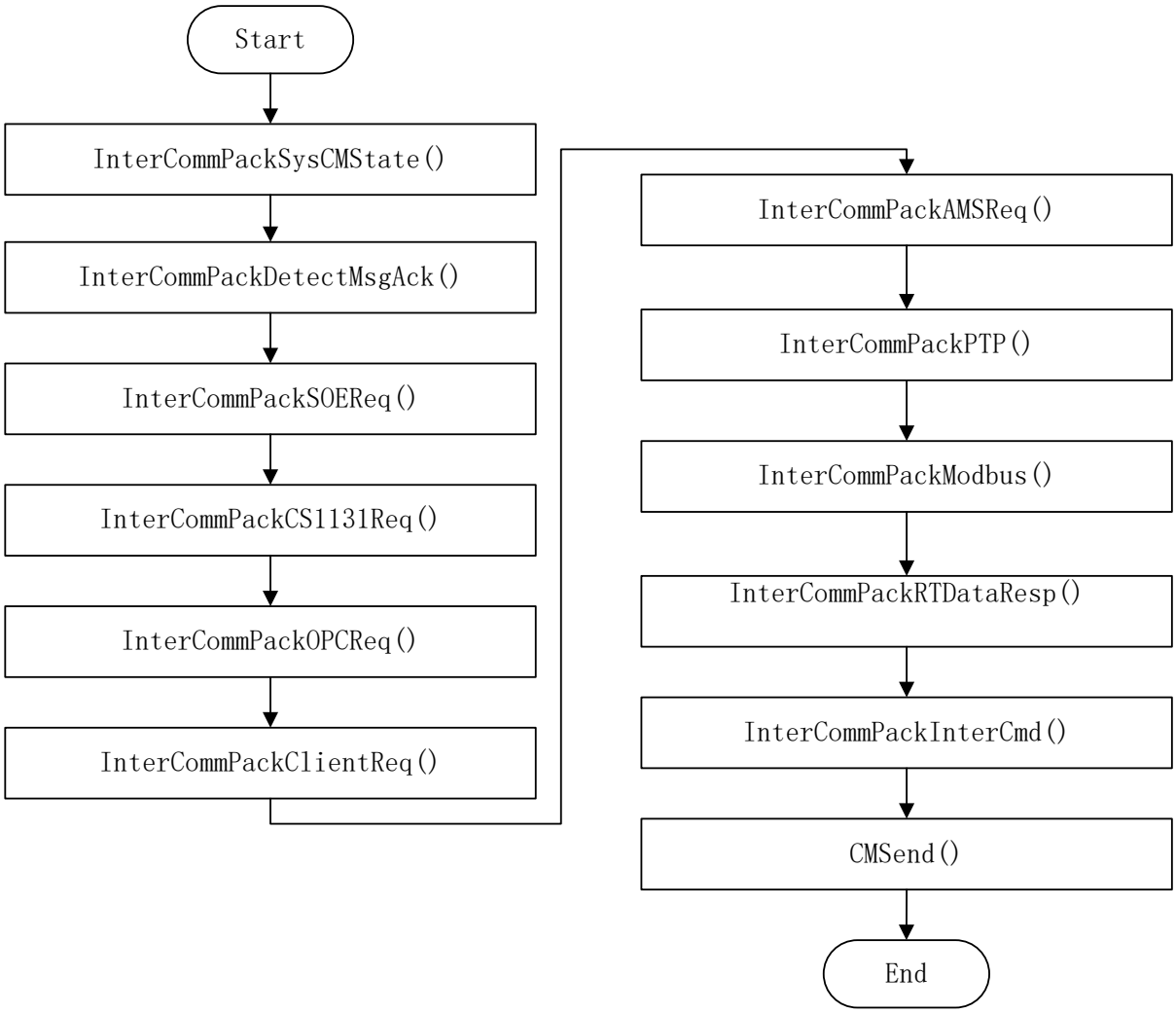

Figure4-4 send data processing flow

图 4-4 发送数据处理流程图

——以下无正文