Document Title: PM_FW safety protocol module design description of Safety Control System

Document Number: 15-Q04-000102

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 PM_FW 安全协议模块设计说明书

文件编号：15-Q04-000102

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 safety protocol function of PM_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 PM_FW, and is the input for the follow-up coding phase.

本文档描述了安全控制系统中 PM_FW 安全协议模块的设计方案。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 PM_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 Abbreviations 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 safety protocol module (marked red) in the software hierarchy is shown below.

安全协议模块（标红）在软件层次中的位置如下图所示。

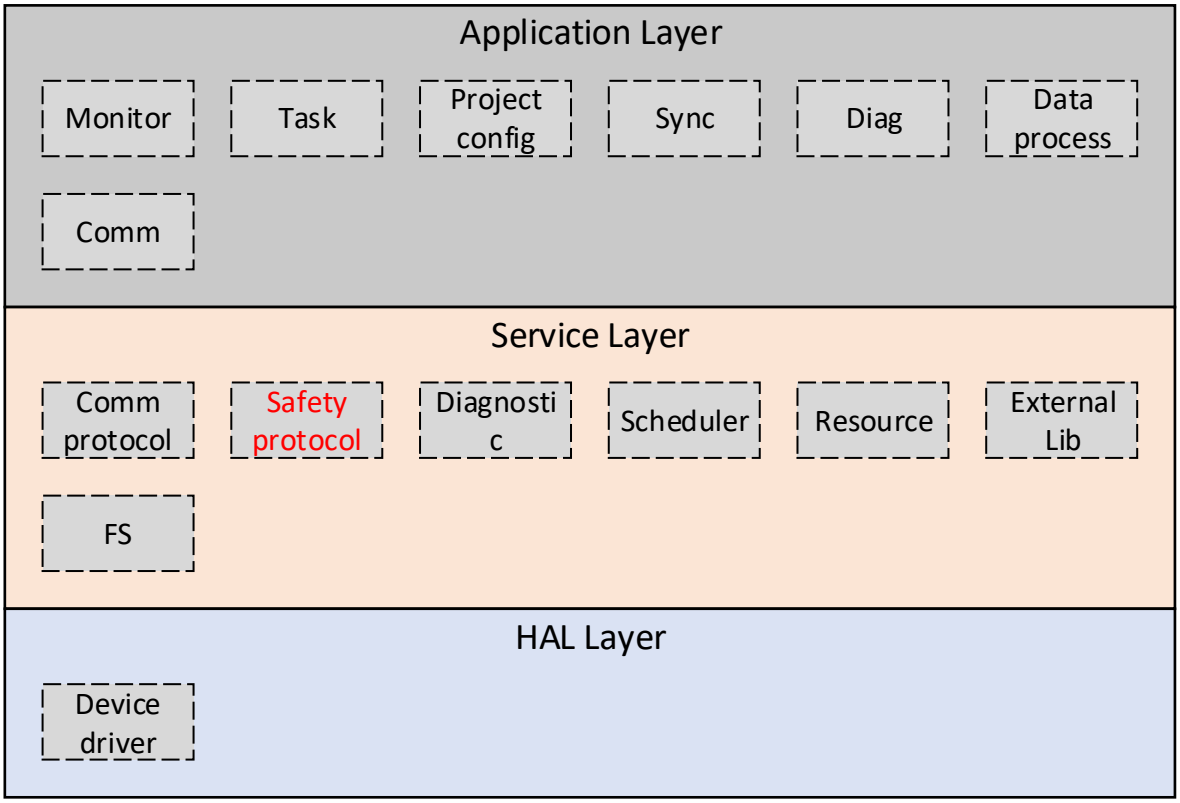

Figure 2-1 the location of the safety protocol module

图 2-1 模块位置

Safety protocol module is mainly used to process safety communication service.

安全协议模块主要用于实现安全通讯服务的处理。

3 P2P safety communication module 控制站间安全通讯模块

3.1 Module design 模块设计

3.1.1 Function description 功能描述

The main functions are as follows:

主要功能如下：

- Initialization;

  初始化；

- Decode request message;

  解析请求帧；

- Make response message;

  构造应答帧；

- Make request message;

  构造请求帧；

- Decode response message.

  解析应答帧。

3.1.2 Design concept 设计思路

3.1.2.1 Decode request message 解析请求帧

Check if exists communication error and update status information. If no error, copy safety data, set its length, and do the recovery operation. If has error, handle according to the redundant and check results:

- Redundant: check if the request message is repeated, if repeated, reset status information, otherwise handle communication error;
- Non-redundant: handle communication error.

检查是否存在通讯错误并更新状态信息。如果无错误，则拷贝安全数据，设置其长度并进行恢复操作。如果存在错误，则根据冗余信息和检查结果进行处理：

- 冗余通信：判断请求帧是否重复，若重复，则复位状态信息，否则进行通信错误处理；
- 非冗余通信：进行通信错误处理。

3.1.2.2 Make response message 构造应答帧

Make response message according to the communication status and safety protocol.

根据通信状态和安全协议构造应答帧。

3.1.2.3 Make request message 构造请求帧

Make request message according to the safety protocol, and update the send sequence number.

根据安全协议构造请求帧，并更新发送序号。

3.1.2.4 Decode response message 解析应答帧

Check if exists communication error. If no error, do the recovery operation, otherwise handle according to the redundant and check results:

- Redundant: check if the response message is repeated, if repeated, discard it, otherwise handle communication error;
- Non-redundant: handle communication error.

检查是否存在通讯错误。如果无错误，则进行恢复操作，否则根据冗余信息和检查结果进行处理：

- 冗余通信：判断应答帧是否重复，若重复，则丢弃，否则进行通信错误处理；
- 非冗余通信：进行通信错误处理。

3.1.3 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 P2PCommInit(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 P2PHandleRecvReqTO(uint8_t ucSendStaID)

Input argument 输入参数	Output argument 输出参数	Description 描述
ucSendStaID: Send control station ID 发送控制站 ID。	If handle successfully. 是否处理成功。	Receive request timeout handling. 接收请求超时处理。

3. bool_t P2PDecodeReq(uint8_t ucSendStaID, bool_t bRedundant, uint8_t pucReq[], uint16_t

usReqLen, ReqDecodeResultInfo_t *pstResult)

Input argument 输入参数	Output argument 输出参数	Description 描述
ucSendStaID: Send control station ID 发送控制站 ID； bRedundant: Redundant communication flag 冗余通信标志； pucReq: Request message 请求帧； usReqLen: Request message length 请求帧长度。	pstResult: Decode result 解析结果； Return: If decode successfully 是否解析成功。	Decode request message. 解析请求帧。

4. bool_t P2PMakeResp(uint8_t ucSendStaID, uint8_t pucResp[], uint16_t *pusRespLen)

Input argument 输入参数	Output argument 输出参数	Description 描述
ucSendStaID: Send control station ID 发送控制站 ID； pucResp: Response buffer 应答缓冲区； pusRespLen: Response buffer length 应答缓冲区长度。	pusRespLen: Response message length 应答帧长度； Return: If make successfully 是否构造成功。	Make response message. 构造应答帧。

5. bool_t P2PHandleSendReqTO(uint8_t ucRecvStaID, bool_t bTimeout)

Input argument 输入参数	Output argument 输出参数	Description 描述
ucRecvStaID: Receive control station ID 接收控制站 ID； bTimeout: Timeout flag 超时标志。	If handle successfully. 是否处理成功。	Send request timeout handling. 发送请求超时处理。

6. bool_t P2PMakeReq(uint8_t ucRecvStaID, uint8_t const pucData[], uint16_t usDataLen, uint8_t pucReq[], uint16_t *pusReqLen)

Input argument 输入参数	Output argument 输出参数	Description 描述
ucRecvStaID: Receive control station ID 接收控制站 ID； pucData: Data 数据； usDataLen: Data length 数据长度； pucReq: Request buffer 请求缓冲区； pusReqLen: Request buffer length 请求缓冲区长度。	pusReqLen: Request message length 请求帧长度； Return: If make successfully 是否构造成功。	Make request message. 构造请求帧。

7. bool_t P2PHandleRespTO(uint8_t ucRecvStaID)

Input argument 输入参数	Output argument 输出参数	Description 描述
ucRecvStaID: Receive control station ID 接收控制站 ID。	If handle successfully. 是否处理成功。	Response timeout handling. 应答超时处理。

8. bool_t P2PDecodeResp(uint8_t ucRecvStaID, bool_t bRedundant, uint8_t pucResp[], uint16_t usRespLen, RespDecodeResultInfo_t *pstResult)

Input argument 输入参数	Output argument 输出参数	Description 描述
ucRecvStaID: Receive control station ID 接收控制站 ID； bRedundant: Redundant communication flag 冗余通信标志； pucResp: Response message 应答帧； usRespLen: Response message length 应答帧长度。	pstResult: Decode result 解析结果； Return: If decode successfully 是否解析成功。	Decode response message. 解析应答帧。

3.1.4 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.	static uint8_t	s_ucTolCnt	Tolerance count. 可容忍次数。
2.	static uint8_t	s_ucTolThr	Tolerance threshold. 容忍门限值。
3.	static P2PRecvReqCommInfo_t	s_stP2PRecvInfo	Receive information. 接收信息。
4.	static P2PSendReqCommInfo_t	s_stP2PSendInfo	Send information. 发送信息。

3.1.5 Data structure 数据结构

1. P2P receive request communication information structure

```
typedef struct P2PRecvReqCommInfoTag
{
    CommErrorInfo_t stErrInfo;
    uint16_t usRecvSeqNum;
    uint16_t usStatusInfo;
}P2PRecvReqCommInfo_t;
```

2. P2P send request communication information structure

```
typedef struct P2PSendReqCommInfoTag
{
    CommErrorInfo_t stErrInfo;
    uint16_t usSendSeqNum;
    uint16_t usLastResSeqNum;
    uint8_t ucCtrlByte;
    uint8_t ucReserve[3];
}P2PSendReqCommInfo_t;
```

3. Station safety data information structure

```
typedef struct StaSafetyDataInfoTag
{
    bool_t bValid;
```

```
    uint8_t ucReqStatus;
    uint16_t usReqSendSeq;
    uint16_t usDataLen;
    uint8_t ucBuff[MAX_P2PSAFETY_DATA_LEN];
}StaSafetyDataInfo_t;
```

4. Request decode information structure

```
typedef struct ReqDecodeInfoTag
{
    bool_t bRepeat;
    uint8_t ucReserve[3];
    StaSafetyDataInfo_t stPMADataInfo;
    StaSafetyDataInfo_t stPMBDataInfo;
    StaSafetyDataInfo_t stPMCDataInfo;
}ReqDecodeInfo_t;
```

5. Request decode result information structure

```
typedef struct ReqDecodeResultInfoTag
{
    bool_t bRepeat;
    uint8_t ucReserve[3];
    PMController_t emSelPMID;
    uint16_t usDataLen;
    uint8_t ucBuff[MAX_P2PSAFETY_DATA_LEN];
}ReqDecodeResultInfo_t;
```

6. Response decode result information structure

```
typedef struct RespDecodeResultInfoTag
{
    bool_t bRepeat;
    uint8_t ucReserve[3];
}RespDecodeResultInfo_t;
```

7. Communication error information structure

```
typedef struct CommErrorInfoTag
{
    uint16_t usErrCnt;
    bool_t bTolFlag;
    bool_t bErrFlag;
}CommErrorInfo_t;
```

8. Station response head information structure

```
typedef struct StaRespHeadInfoTag
{
    uint16_t usSeq;
    uint8_t ucSrcAddr;
    uint8_t ucDstAddr;
    uint16_t usStatus;
    uint16_t usSDataLen;
}StaRespHeadInfo_t;
```

3.1.6 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-PM-SP_SafR_SecR_A_001	Module initialization 模块初始化
SWDD-PM-SP_SafR_SecR_A_002	Decode request message 解析请求帧
SWDD-PM-SP_SafR_SecR_A_003	Receive request timeout handling 接收请求超时处理
SWDD-PM-SP_SafR_SecR_A_004	Make response message 构造应答帧
SWDD-PM-SP_SafR_SecR_A_005	Make request message 构造请求帧
SWDD-PM-SP_SafR_SecR_A_006	Send request timeout handling 发送请求超时处理
SWDD-PM-SP_SafR_SecR_A_007	Decode response message 解析应答帧
SWDD-PM-SP_SafR_SecR_A_008	Response timeout handling 应答超时处理
SWDD-PM-SP_SafR_SecR_A_009	Communication recovery handling 通信恢复处理
SWDD-PM-SP_SafR_SecR_A_010	Communication error handling 通信错误处理

3.2 Design of sun-function 子功能设计

3.2.1 Module initialization 模块初始化

SWDD-PM-SP_SafR_SecR_A_001

3.2.1.1 P2PCommInit

3.2.1.1.1 Function Description 功能描述

This function completes initialization of module.

该函数完成模块的初始化。

3.2.1.1.2 Argument Description 参数说明

- Definition 函数定义

void P2PCommInit(void)

- Input argument 输入参数

No.

无。

- Output argument 输出参数

No.

无。

3.2.1.1.3 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。

3.2.2 Decode request message 解析请求帧

SWDD-PM-SP_SafR_SecR_A_002

3.2.2.1 P2PDecodeReq

3.2.2.1.1 Function Description 功能描述

This function is used to decode request message.

该函数用于解析请求帧。

3.2.2.1.2 Argument Description 参数说明

- Definition 函数定义

bool_t P2PDecodeReq(uint8_t ucSendStaID, bool_t bRedundant, uint8_t pucReq[], uint16_t usReqLen, ReqDecodeResultInfo_t *pstResult)

- Input argument 输入参数

ucSendStaID: Send control station ID 发送控制站 ID；

bRedundant: Redundant communication flag 冗余通信标志；

pucReq: Request message 请求帧；

usReqLen: Request message length 请求帧长度。

- Output argument 输出参数

pstResult: Decode result 解析结果；

Return: If decode successfully 是否解析成功。

3.2.2.1.3 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 communication error: check if redundant data are the same between each other, if CRC, address and sequence number of the request message are correct, and if the PM information in the additional field is correct;

   通信错误检查：检查冗余数据是否相同，请求帧的 CRC、地址和序号是否正确和附加信息中的 PM 信息是否正确；

2. Update status information according to the check results;

   根据检查结果更新状态信息；

3. Check if has valid request frame: if has, update receive sequence number and do the communication recovery operation, see section 3.2.9 for details, otherwise enter step 4;

   检查是否存在有效的请求帧：如果存在，则更新接收序号并进行通信恢复操作-详见 3.2.9 节，否则进入步骤 4；

4. Check if all sequences are error: if yes, enter step 5, otherwise reset receive sequence and do the communication error operation, see section 3.2.10 for details;

   检查是否均是序号错：如果是，则进入步骤 5，否则复位接收序号并进行通信错误处理-详见 3.2.10 节；

5. Check if all sequences are equal: if yes, enter step 6, otherwise reset receive sequence and do the communication error operation, see section 3.2.10 for details;

   检查是否均是序号错：如果是，则进入步骤 6，否则复位接收序号并进行通信错误处理-详见 3.2.10 节；

6. Check if redundant: if yes, enter step 7, otherwise update receive sequence and do the communication error operation, see section 3.2.10 for details;

   检查是否冗余：如果是，则进入步骤 7，否则更新接收序号并进行通信错误处理-详

见 3.2.10 节；

7. Check if repeated: if yes, reset status information, otherwise update receive sequence and do the communication error operation, see section 3.2.10 for details.

   检查是否冗余：如果是，则复位状态信息，否则更新接收序号并进行通信错误处理-详见 3.2.10 节。

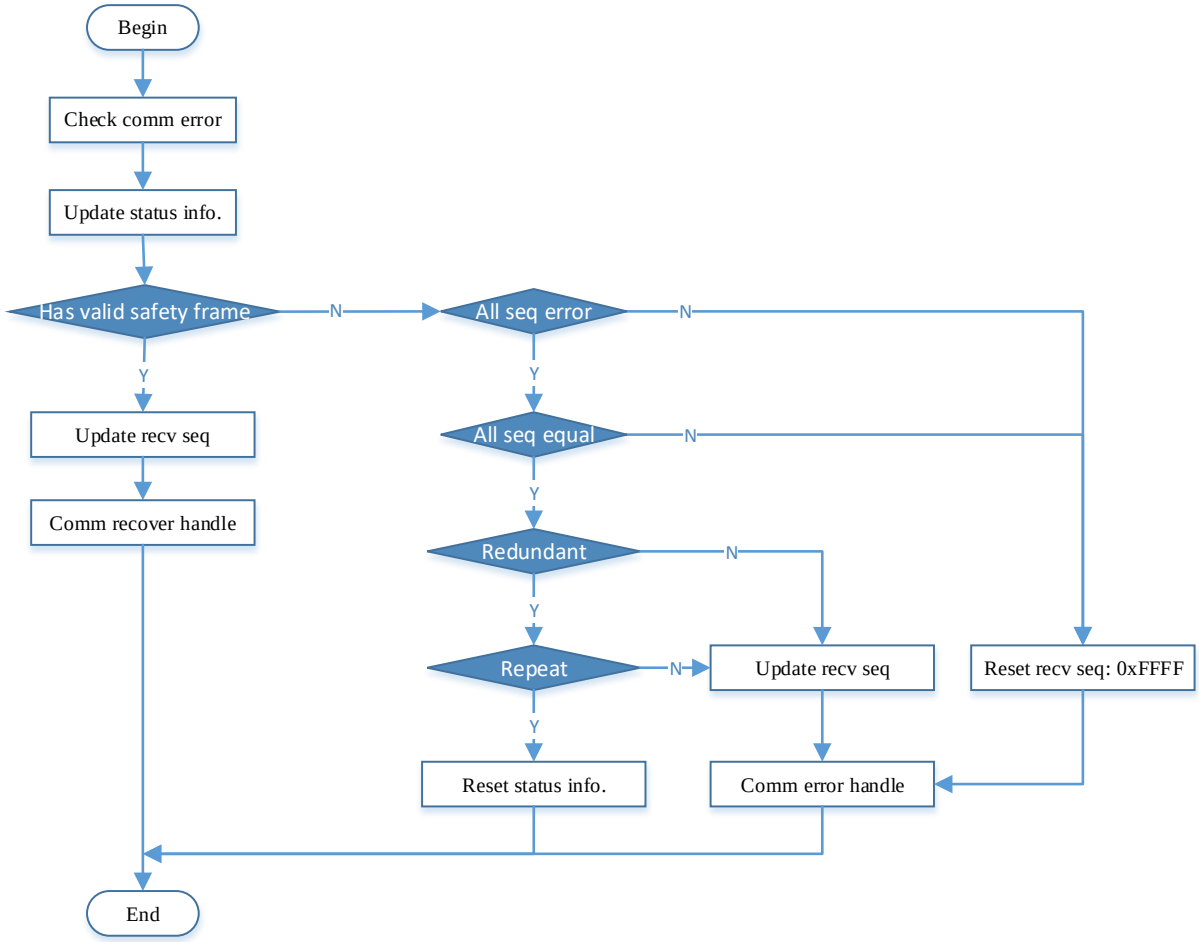

Figure 3-1 decode request message

图 3-1 解析请求帧

3.2.3 Receive request timeout handling 接收请求超时处理

SWDD-PM-SP_SafR_SecR_A_003

3.2.3.1 P2PHandleRecvReqTO

3.2.3.1.1 Function Description 功能描述

This function is used to handle receive request timeout.

该函数用于处理接收请求超时。

3.2.3.1.2 Argument Description 参数说明

- Definition 函数定义

bool_t P2PHandleRecvReqTO(uint8_t ucSendStaID)

- Input argument 输入参数

ucSendStaID: Send control station ID 发送控制站 ID。

- Output argument 输出参数

If handle successfully.

是否处理成功。

3.2.3.1.3 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. Update status information: Set receive timeout;

   更新状态信息：置接收超时；

2. Communication error handling: see section 3.2.10 for details.

   通信错误处理：详见 3.2.10 节。

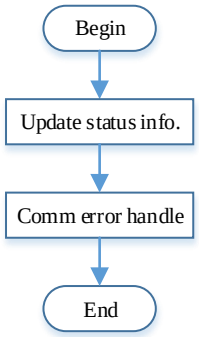

Figure 3-2 handle receive request timeout
图 3-2 处理接收请求超时

3.2.4 Make response message 构造应答帧

SWDD-PM-SP_SafR_SecR_A_004

3.2.4.1 P2PMakeResp

3.2.4.1.1 Function Description 功能描述

This function is used to make response message.

该函数用于构造应答帧。

3.2.4.1.2 Argument Description 参数说明

- Definition 函数定义

bool_t P2PMakeResp(uint8_t ucSendStaID, uint8_t pucResp[], uint16_t *pusRespLen)

- Input argument 输入参数

ucSendStaID: Send control station ID 发送控制站 ID；

pucResp: Response buffer 应答缓冲区；

pusRespLen: Response buffer length 应答缓冲区长度。

- Output argument 输出参数

pusRespLen: Response message length 应答帧长度；

Return: If make successfully 是否构造成功。

3.2.4.1.3 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。

3.2.5 Make request message 构造请求帧

SWDD-PM-SP_SafR_SecR_A_005

3.2.5.1 P2PMakeReq

3.2.5.1.1 Function Description 功能描述

This function is used to make request message.

该函数用于构造请求帧。

3.2.5.1.2 Argument Description 参数说明

- Definition 函数定义

bool_t P2PMakeReq(uint8_t ucRecvStaID, uint8_t const pucData[], uint16_t usDataLen,
uint8_t pucReq[], uint16_t *pusReqLen)

- Input argument 输入参数

ucRecvStaID: Receive control station ID 接收控制站 ID；

pucData: Data 数据；

usDataLen: Data length 数据长度；

pucReq: Request buffer 请求缓冲区；

pusReqLen: Request buffer length 请求缓冲区长度。

- Output argument 输出参数

pusReqLen: Request message length 请求帧长度；

Return: If make successfully 是否构造成功。

3.2.5.1.3 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。

3.2.6 Send request timeout handling 发送请求超时处理

SWDD-PM-SP_SafR_SecR_A_006

3.2.6.1 P2PHandleSendReqTO

3.2.6.1.1 Function Description 功能描述

This function is used to handle send request timeout: set send timeout flag.

该函数用于处理发送请求超时：设置发送超时标志位。

3.2.6.1.2 Argument Description 参数说明

- Definition 函数定义

bool_t P2PHandleSendReqTO(uint8_t ucRecvStaID, bool_t bTimeout)

- Input argument 输入参数

ucRecvStaID: Receive control station ID 接收控制站 ID；
bTimeout: Timeout flag 超时标志。

- Output argument 输出参数

If handle successfully.

是否处理成功。

3.2.6.1.3 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。

3.2.7 Decode response message 解析应答帧

SWDD-PM-SP_SafR_SecR_A_007

3.2.7.1 P2PDecodeResp

3.2.7.1.1 Function Description 功能描述

This function is used to decode response message.

该函数用于解析应答帧。

3.2.7.1.2 Argument Description 参数说明

- Definition 函数定义

bool_t P2PDecodeResp(uint8_t ucRecvStaID, bool_t bRedundant, uint8_t pucResp[],

uint16_t usRespLen, RespDecodeResultInfo_t *pstResult)

- Input argument 输入参数

ucRecvStaID: Receive control station ID 接收控制站 ID；
bRedundant: Redundant communication flag 冗余通信标志；
pucResp: Response message 应答帧；
usRespLen: Response message length 应答帧长度。

- Output argument 输出参数

pstResult: Decode result 解析结果；
Return: If decode successfully 是否解析成功。

3.2.7.1.3 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 communication error: check if redundant data are the same between each other, and if CRC, address and sequence number of the response message are correct;

   通信错误检查：检查冗余数据是否相同，应答帧的 CRC、地址和序号是否正确；

2. Check if no communication error: if no error, enter step 3, otherwise enter step 5;

   检查是否无通信错误：如果是，则进入步骤 3，否则进入步骤 5；

3. Check if repeat: if no, do the communication recovery operation, see section 3.2.9 for details;

   检查是否重复：如果否，则进行通信恢复操作-详见 3.2.9 节；

4. Update last response sequence number: set using the sequence number of the current response message;

   更新上一应答序号：设置为当前应答帧的序号；

5. Check if sequence error: if yes, enter step 6, otherwise do the communication error operation, see section 3.2.10 for details;

   检查是否序号错：如果是，则进入步骤 6，否则进行通信错误处理-详见 3.2.10 节；

6. Check if repeated: if yes, update last response sequence as step 4, otherwise enter step 7;

   检查是否重复：如果是，则更新上一应答序号-同步骤 4，否则进入步骤 7；

7. Handle local send sequence number, see section 3.2.7.2 for details, do the communication error operation, see section 3.2.10 for details, and update last response sequence number as step 4.

更新本地发送序号-详见 3.2.7.2 节，进行通信错误处理-详见 3.2.10 节，并更新上一应答序号-同步骤 4。

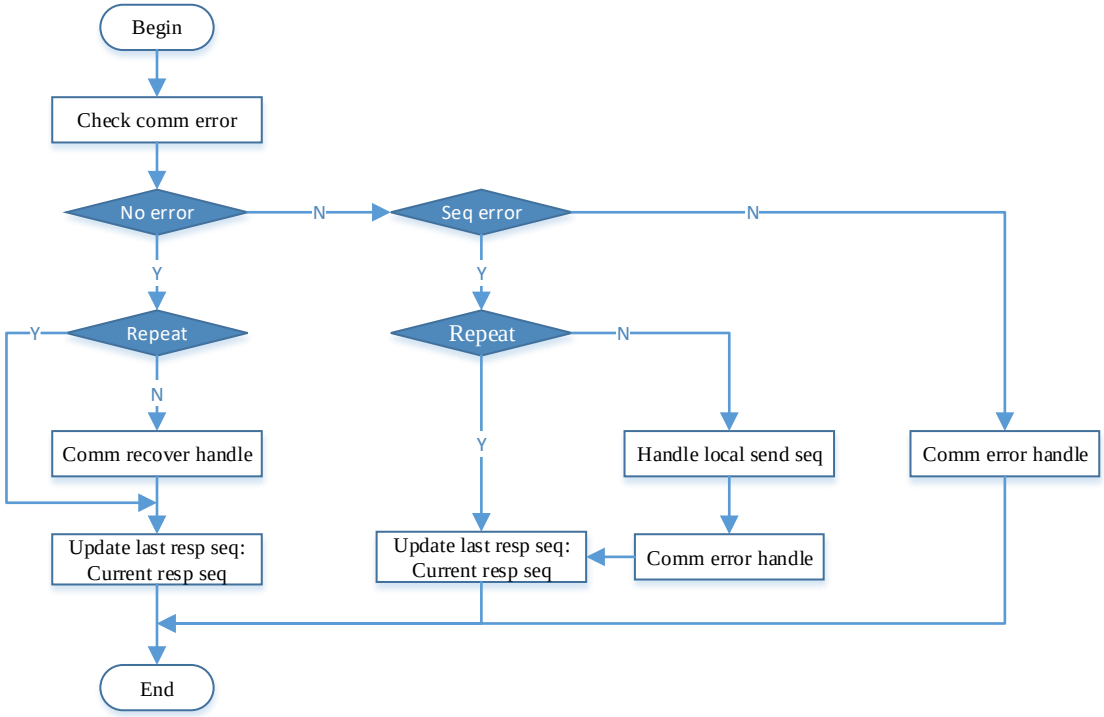

Figure 3-3 decode response message

图 3-3 解析应答帧

3.2.7.2 HandleLocalSendSQ

3.2.7.2.1 Function Description 功能描述

This function is used to handle local send sequence.

该函数用于处理本地发送序号。

3.2.7.2.2 Argument Description 参数说明

- Definition 函数定义

void HandleLocalSendSQ(StaRespHeadInfo_t const *pstHead, P2PSendReqCommInfo_t *pstStaInfo)

- Input argument 输入参数

pstHead: Response message header 应答帧头；

pstStaInfo: Send request communication information 发送请求通信信息。

- Output argument 输出参数

No.

无。

3.2.7.2.3 Processing flow 处理流程

The processing flow is shown below:

流程如下图所示：

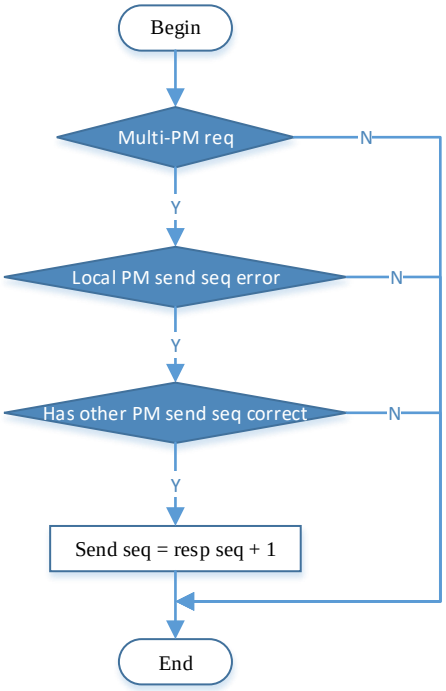

Figure 3-4 handle local send sequence

图 3-4 处理本地发送序号

3.2.8 Response timeout handling 应答超时处理

SWDD-PM-SP_SafR_SecR_A_008

3.2.8.1 P2PHandleRespTO

3.2.8.1.1 Function Description 功能描述

This function is used to handle response timeout: do the communication error operation, see section 3.2.10 for details.

该函数用于处理应答超时：进行通信错误处理-详见 3.2.10 节。

3.2.8.1.2 Argument Description 参数说明

- Definition 函数定义

bool_t P2PHandleRespTO(uint8_t ucRecvStaID)

- Input argument 输入参数

ucRecvStaID: Receive control station ID 接收控制站 ID。

- Output argument 输出参数

If handle successfully.

是否处理成功。

3.2.8.1.3 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。

3.2.9 Communication recovery handling 通信恢复处理

SWDD-PM-SP_SafR_SecR_A_009

3.2.9.1 HandleSafetyCommRecover

3.2.9.1.1 Function Description 功能描述

This function is used to handle communication recovery.

该函数用于通信恢复处理。

3.2.9.1.2 Argument Description 参数说明

- Definition 函数定义

void HandleSafetyCommRecover(CommErrorInfo_t *pstErrInfo)

- Input argument 输入参数

pstErrInfo: Error information 错误信息。

- Output argument 输出参数

No.

无。

3.2.9.1.3 Processing flow 处理流程

The processing flow is shown below:

流程如下图所示：

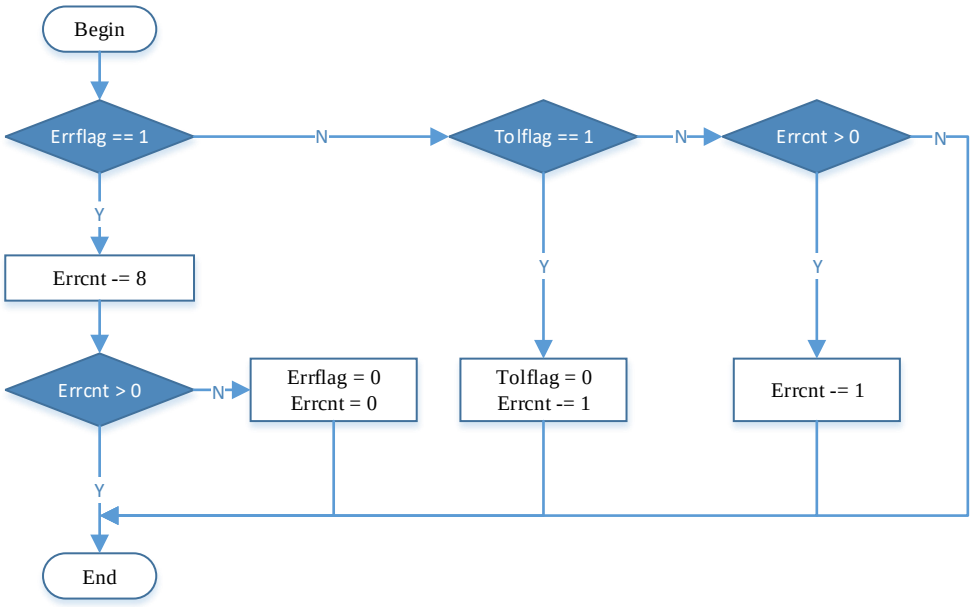

Figure 3-5 handle communication recover

图 3-5 通信恢复处理

3.2.10 Communication error handling 通信错误处理

SWDD-PM-SP_SafR_SecR_A_010

3.2.10.1 HandleSafetyCommError

3.2.10.1.1 Function Description 功能描述

This function is used to handle communication error.

该函数用于通信错误处理。

3.2.10.1.2 Argument Description 参数说明

- Definition 函数定义

void HandleSafetyCommError(CommErrorInfo_t *pstErrInfo, uint8_t ucTolThr)

- Input argument 输入参数

pstErrInfo: Error information 错误信息；

ucTolThr: Tolerate threshold 容忍门限值。

- Output argument 输出参数

No.

无。

3.2.10.1.3 Processing flow 处理流程

The processing flow is shown below:

流程如下图所示：

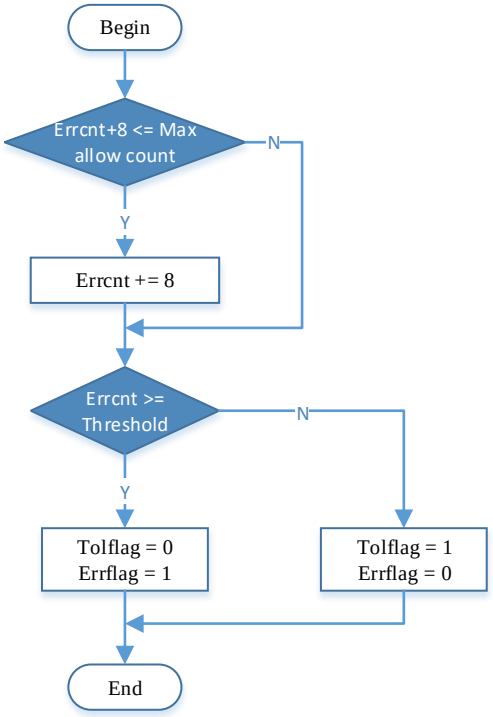

Figure 3-6 handle communication error

图 3-6 通信错误处理

4 PM safety communication module PM 间安全通讯模块

4.1 Module design 模块设计

4.1.1 Function description 功能描述

The main functions are as follows:

主要功能如下：

- Initialization;

  初始化；

- Decode safety message;

  解析安全帧；

- Make safety message;

  构造安全帧；

4.1.2 Design concept 设计思路

4.1.2.1 Decode safety message 解析安全帧

Check if exists communication error: If no error, copy safety data, set its length, and do the recovery operation, otherwise handle communication error.

检查是否存在通讯错误：如果无错误，则拷贝安全数据，设置其长度并进行恢复操作，否则进行通信错误处理。

4.1.2.2 Make safety message 构造安全帧

Make safety message according to the safety protocol.

根据安全协议构造安全帧。

4.1.3 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 PMCommInit(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 PMDecodeFra(PMController_t emSendPMID, uint8_t const pucFra[], uint16_t usFraLen, uint8_t pucData[], uint16_t *pusDataLen)

Input argument 输入参数	Output argument 输出参数	Description 描述
emSendPMID: Send PM ID 发送 PM ID； pucFra: Safety message 安全帧； usFraLen: Message length 安全帧长度； pucData: Data buffer 数据缓冲区； pusDataLen: Data buffer length 数据缓冲区长度。	pusDataLen: Data length 数据长度； Return: If decode successfully 是否解析成功。	Decode safety message. 解析安全帧。

3. bool_t PMHandleRecvTO(PMController_t emSendPMID)

Input argument 输入参数	Output argument 输出参数	Description 描述
emSendPMID: Send PM ID 发送 PM ID。	If handle successfully. 是否处理成功。	Receive timeout handling. 接收超时处理。

4. bool_t PMMakeFra(PMController_t emRecvPMID, uint8_t const pucData[], uint16_t

usDataLen, uint8_t pucFra[], uint16_t *pusFraLen)

Input argument 输入参数	Output argument 输出参数	Description 描述
emRecvPMID: Receive PM ID 接收 PM ID； pucData: Data to be packaged 将被打包的数据； usDataLen: Data length 数据长度； pucFra: Safety message buffer 安全帧缓冲区； pusFraLen: Message buffer length 安全帧缓冲区长度。	pusFraLen: Safety message length 安全帧长度； Return: If make successfully 是否构造成功。	Make safety message. 构造安全帧。

5. bool_t PMHandleSendTO(PMController_t emRecvPMID, bool_t bTimeout)

Input argument 输入参数	Output argument 输出参数	Description 描述
emRecvPMID: Receive PM ID 接收 PM ID； bTimeout: Timeout flag 超时标志。	If handle successfully. 是否处理成功。	Send timeout handling. 发送超时处理。

4.1.4 Global variable 全局变量

Table 4-1 Global variable list
表 4-1 全局变量列表

No. 序号	Type 变量类型	Name 名称	Description 描述
1.	static uint8_t	s_ucTolCnt	Tolerance count. 可容忍次数。
2.	static uint8_t	s_ucTolThr	Tolerance threshold. 容忍门限值。
3.	static PMRecvInfo_t	s_stPMRecvInfo	Receive information. 接收信息。
4.	static PMSendInfo_t	s_stPMSendInfo	Send information. 发送信息。

4.1.5 Data structure 数据结构

1. PM receive information structure

```
typedef struct PMRecvInfoTag
{
    CommErrorInfo_t stErrInfo;
    uint16_t usRecvSeqNum;
    uint8_t ucReserve[2];
}PMRecvInfo_t;
```

2. PM send information structure

```
typedef struct PMSendInfoTag
{
    uint16_t usSendSeqNum;
    uint8_t ucReserve[2];
}PMSendInfo_t;
```

3. PM safety head information structure

```
typedef struct PMSafetyHeadInfoTag
{
    uint8_t ucSrcPMID;
    uint8_t ucDstPMID;
    uint16_t usSeq;
    uint32_t uiSDataLen;
}PMSafetyHeadInfo_t;
```

4.1.6 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 4-2 Sub function list

表 4-2 子功能列表

Sub function No. 子功能编号	Function description 功能描述
SWDD-PM-SP_SafR_NSecR_A_011	Module initialization 模块初始化
SWDD-PM-SP_SafR_SecR_A_012	Decode safety message 解析安全帧
SWDD-PM-SP_SafR_NSecR_A_013	Receive timeout handling 接收超时处理

SWDD-PM-SP_SafR_NSecR_A_014	Make safety message 构造安全帧
SWDD-PM-SP_SafR_NSecR_A_015	Send timeout handling 发送超时处理

4.2 Design of sun-function 子功能设计

4.2.1 Module initialization 模块初始化

SWDD-PM-SP_SafR_NSecR_A_011

4.2.1.1 PMCommInit

4.2.1.1.1 Function Description 功能描述

This function completes initialization of module.

该函数完成模块的初始化。

4.2.1.1.2 Argument Description 参数说明

- Definition 函数定义

void PMCommInit(void)

- Input argument 输入参数

No.

无。

- Output argument 输出参数

No.

无。

4.2.1.1.3 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。

4.2.2 Decode safety message 解析安全帧

SWDD-PM-SP_SafR_SecR_A_012

4.2.2.1 PMDecodeFra

4.2.2.1.1 Function Description 功能描述

This function is used to decode safety message.

该函数用于解析安全帧。

4.2.2.1.2 Argument Description 参数说明

- Definition 函数定义

bool_t PMDecodeFra(PMController_t emSendPMID, uint8_t const pucFra[], uint16_t usFraLen, uint8_t pucData[], uint16_t *pusDataLen)

- Input argument 输入参数

emSendPMID: Send PM ID 发送 PM ID；
pucFra: Safety message 安全帧；
usFraLen: Message length 安全帧长度；
pucData: Data buffer 数据缓冲区；
pusDataLen: Data buffer length 数据缓冲区长度。

- Output argument 输出参数

pusDataLen: Data length 数据长度；
Return: If decode successfully 是否解析成功。

4.2.2.1.3 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 communication error: check if CRC, address and sequence number of the safety message are correct;

   通信错误检查：检查安全帧的 CRC、地址和序号是否正确；

2. Check if no error: if no error, update receive sequence number and do the communication recovery operation, see section 3.2.9 for details, otherwise enter step 3;

   检查是否无错误：如果是，则更新接收序号并进行通信恢复操作-详见 3.2.9 节，否则进入步骤 3；

3. Check if sequence error: if yes, enter step 4, otherwise do the communication error operation, see section 3.2.10 for details;

   检查是否序号错：如果是，则进入步骤 4，否则进行通信错误处理-详见 3.2.10 节；

4. Update receive sequence number and do the communication error operation, see section 3.2.10 for details.

   更新接收序号并进行通信错误处理-详见 3.2.10 节。

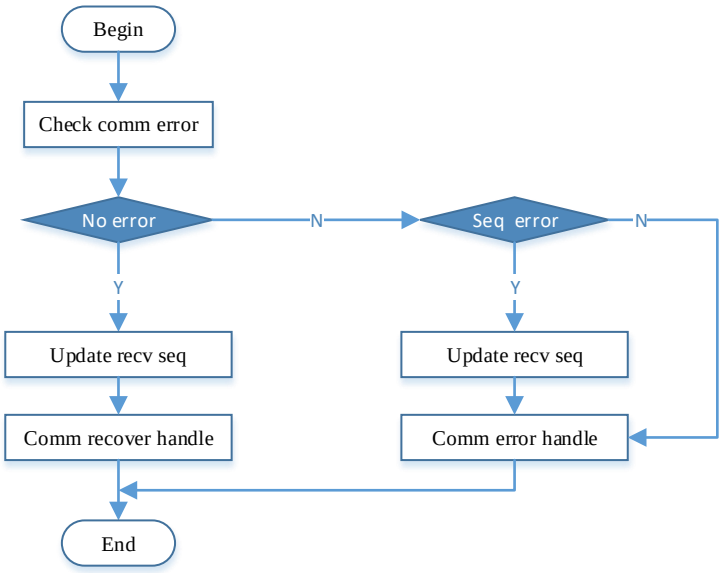

Figure 4-1 decode safety message
图 4-1 解析安全帧

4.2.3 Receive timeout handling 接收超时处理

SWDD-PM-SP_SafR_NSecR_A_013

4.2.3.1 PMHandleRecvTO

4.2.3.1.1 Function Description 功能描述

This function is used to handle receive timeout: do the communication error operation, see section 3.2.10 for details.

该函数用于处理接收超时：进行通信错误处理-详见 3.2.10 节。

4.2.3.1.2 Argument Description 参数说明

- Definition 函数定义

bool_t PMHandleRecvTO(PMController_t emSendPMID)

- Input argument 输入参数

emSendPMID: Send PM ID 发送 PM ID。

- Output argument 输出参数

If handle successfully.

是否处理成功。

4.2.3.1.3 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。

4.2.4 Make safety message 构造安全帧

SWDD-PM-SP_SafR_NSecR_A_014

4.2.4.1 PMMakeFra

4.2.4.1.1 Function Description 功能描述

This function is used to make safety message.

该函数用于构造安全帧。

4.2.4.1.2 Argument Description 参数说明

- Definition 函数定义

bool_t PMMakeFra(PMController_t emRecvPMID, uint8_t const pucData[], uint16_t usDataLen, uint8_t pucFra[], uint16_t *pusFraLen)

- Input argument 输入参数

emRecvPMID: Receive PM ID 接收 PM ID；
pucData: Data to be packaged 将被打包的数据；
usDataLen: Data length 数据长度；
pucFra: Safety message buffer 安全帧缓冲区；
pusFraLen: Message buffer length 安全帧缓冲区长度。

- Output argument 输出参数

pusFraLen: Safety message length 安全帧长度；
Return: If make successfully 是否构造成功。

4.2.4.1.3 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。

4.2.5 Send timeout handling 发送超时处理

SWDD-PM-SP_SafR_NSecR_A_015

4.2.5.1 PMHandleSendTO

4.2.5.1.1 Function Description 功能描述

This function is used to handle send timeout: set send timeout flag.

该函数用于处理发送超时：设置发送超时标志位。

4.2.5.1.2 Argument Description 参数说明

- Definition 函数定义

bool_t PMHandleSendTO(PMController_t emRecvPMID, bool_t bTimeout)

- Input argument 输入参数

emRecvPMID: Receive PM ID 接收 PM ID；
bTimeout: Timeout flag 超时标志。

- Output argument 输出参数

If handle successfully.

是否处理成功。

4.2.5.1.3 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。

5 PMIO safety communication module PM 与 IO 间安全通讯模块

5.1 Module design 模块设计

5.1.1 Function description 功能描述

The main functions are as follows:

主要功能如下：

- Initialization;

  初始化；

- Make request message;

  构造请求帧；

- Decode response message.

  解析应答帧。

5.1.2 Design concept 设计思路

5.1.2.1 Make request message 构造请求帧

Make request message according to the safety protocol and communication status, and update the send sequence number.

根据安全协议和通信状态构造请求帧，并更新发送序号。

5.1.2.2 Decode response message 解析应答帧

Check if exists communication error: if no error, do the recovery operation, otherwise handle communication error.

检查是否存在通讯错误：如果无错误，则进行恢复操作，否则进行通信错误处理。

5.1.3 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 PMIOCommInit(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 PMIOMakeReq(uint8_t ucIOID, uint8_t const pucData[], uint8_t ucDataLen, bool_t bActiveFV, bool_t bEnableFlg, uint8_t pucReq[], uint16_t *pusReqLen)

Input argument 输入参数	Output argument 输出参数	Description 描述
ucIOID: I/O module ID I/O 模块 ID； pucData: Data to be packaged 将被打包的数据； usDataLen: Data length 数据长度； bActiveFV: ActiveFV flag 使用故障安全值标志； bEnableFlg: Enable IO parameter flag 使能 IO 模块参数标志； pucReq: Request message buffer 请求帧缓冲区； pusReqLen: Message buffer length 请求帧缓冲区长度。	pusReqLen: Request message length 请求帧长度； Return: If make successfully 是否构造成功。	Make request message. 构造请求帧。

3. bool_t PMIOHandleSendTO(uint8_t ucIOID, bool_t bTimeout)

Input argument 输入参数	Output argument 输出参数	Description 描述
ucIOID: I/O module ID I/O 模块 ID； bTimeout: Timeout flag 超时	If handle successfully. 是否处理成功。	Send timeout handling. 发送超时处理。

标志。		

4. bool_t PMIODecodeResp(uint8_t ucIOID, uint8_t pucResp[], uint16_t usRespLen, uint8_t pucData[], uint8_t *pucDataLen, bool_t *pbParOK)

Input argument 输入参数	Output argument 输出参数	Description 描述
ucIOID: I/O module ID I/O 模块 ID; pucResp: Response message 应答帧; usRespLen: Message length 应答帧长度; pucData: Data buffer 数据缓冲区; pucDataLen: Data buffer length 数据缓冲区长度。	pucDataLen: Data length 数据长度; pbParOK: Parameter ok flag 参数有效标志; Return: If decode successfully 是否解析成功。	Decode response message. 解析应答帧。

5. bool_t PMIOHandleRecvTO(uint8_t ucIOID)

Input argument 输入参数	Output argument 输出参数	Description 描述
ucIOID: I/O module ID I/O 模块 ID。	If handle successfully. 是否处理成功。	Receive timeout handling. 接收超时处理。

5.1.4 Global variable 全局变量

Table 5-1 Global variable list
表 5-1 全局变量列表

No. 序号	Type 变量类型	Name 名称	Description 描述
1.	static uint8_t	s_ucTolCnt	Tolerance count. 可容忍次数。
2.	static uint8_t	s_ucTolThr	Tolerance threshold. 容忍门限值。
3.	static PMIOCommInfo_t	s_stPMIOCommInfo	Communication information. 通信信息。

5.1.5 Data structure 数据结构

1. PM I/O communication information structure

```
typedef struct PMIOCommInfoTag
{
    CommErrorInfo_t stErrInfo;
    uint16_t usSendSeqNum;
    uint8_t ucCtrlByte;
    bool_t bIOModFailure;
    bool_t bFVActived;
    bool_t bNewIPara;
}PMIOCommInfo_t;
```

2. PM I/O request head information structure

```
typedef struct PMIOReqHeadInfoTag
{
    uint16_t usPMID;
    uint16_t usIOID;
    uint16_t usSeq;
    uint8_t ucCtrlByte;
    uint8_t ucSDataLen;
}PMIOReqHeadInfo_t;
```

3. PM I/O response head information structure

```
typedef struct PMIORespHeadInfoTag
{
    uint16_t usIOID;
    uint16_t usPMID;
    uint16_t usSeq;
    uint8_t ucStatusByte;
    uint8_t ucSDataLen;
}PMIORespHeadInfo_t;
```

5.1.6 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 5-2 Sub function list

表 5-2 子功能列表

Sub function No. 子功能编号	Function description 功能描述
SWDD-PM-SP_SafR_SecR_A_016	Module initialization 模块初始化

SWDD-PM-SP_SafR_SecR_A_017	Make request message 构造请求帧
SWDD-PM-SP_SafR_SecR_A_018	Send request timeout handling 发送请求超时处理
SWDD-PM-SP_SafR_SecR_A_019	Decode response message 解析应答帧
SWDD-PM-SP_SafR_SecR_A_020	Response timeout handling 应答超时处理

5.2 Design of sun-function 子功能设计

5.2.1 Module initialization 模块初始化

SWDD-PM-SP_SafR_SecR_A_016

5.2.1.1 PMIOCommInit

5.2.1.1.1 Function Description 功能描述

This function completes initialization of module.

该函数完成模块的初始化。

5.2.1.1.2 Argument Description 参数说明

- Definition 函数定义

void PMIOCommInit(void)

- Input argument 输入参数

No.

无。

- Output argument 输出参数

No.

无。

5.2.1.1.3 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。

5.2.2 Make request message 构造请求帧

SWDD-PM-SP_SafR_SecR_A_017

5.2.2.1 PMIOMakeReq

5.2.2.1.1 Function Description 功能描述

This function is used to make request message.

该函数用于构造请求帧。

5.2.2.1.2 Argument Description 参数说明

- Definition 函数定义

bool_t PMIOMakeReq(uint8_t ucIOID, uint8_t const pucData[], uint8_t ucDataLen, bool_t bActiveFV, bool_t bEnableFlg, uint8_t pucReq[], uint16_t *pusReqLen)

- Input argument 输入参数

ucIOID: I/O module ID I/O 模块 ID；
pucData: Data to be packaged 将被打包的数据；
usDataLen: Data length 数据长度；
bActiveFV: ActiveFV flag 使用故障安全值标志；
bEnableFlg: Enable IO parameter flag 使能 IO 模块参数标志；
pucReq: Request message buffer 请求帧缓冲区；
pusReqLen: Message buffer length 请求帧缓冲区长度。

- Output argument 输出参数

pusReqLen: Request message length 请求帧长度；
Return: If make successfully 是否构造成功。

5.2.2.1.3 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。

5.2.3 Send request timeout handling 发送请求超时处理

SWDD-PM-SP_SafR_SecR_A_018

5.2.3.1 PMIOHandleSendTO

5.2.3.1.1 Function Description 功能描述

This function is used to handle send request timeout: set send timeout flag.

该函数用于处理发送请求超时：设置发送超时标志位。

5.2.3.1.2 Argument Description 参数说明

- Definition 函数定义

bool_t PMIOHandleSendTO(uint8_t ucIOID, bool_t bTimeout)

- Input argument 输入参数

ucIOID: I/O module ID I/O 模块 ID；
bTimeout: Timeout flag 超时标志。

- Output argument 输出参数

If handle successfully.

是否处理成功。

5.2.3.1.3 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。

5.2.4 Decode response message 解析应答帧

SWDD-PM-SP_SafR_SecR_A_019

5.2.4.1 PMIODecodeResp

5.2.4.1.1 Function Description 功能描述

This function is used to decode response message.

该函数用于解析应答帧。

5.2.4.1.2 Argument Description 参数说明

- Definition 函数定义

bool_t PMIODecodeResp(uint8_t ucIOID, uint8_t pucResp[], uint16_t usRespLen, uint8_t pucData[], uint8_t *pucDataLen, bool_t *pbParOK)

- Input argument 输入参数

ucIOID: I/O module ID I/O 模块 ID；
pucResp: Response message 应答帧；
usRespLen: Message length 应答帧长度；
pucData: Data buffer 数据缓冲区；
pucDataLen: Data buffer length 数据缓冲区长度。

- Output argument 输出参数

pucDataLen: Data length 数据长度；
pbParOK: Parameter ok flag 参数有效标志；
Return: If decode successfully 是否解析成功。

5.2.4.1.3 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 communication error: check if redundant data are the same between each other, if CRC, address and sequence number of the response message are correct;

   通信错误检查：检查冗余数据是否相同，应答帧的 CRC、地址和序号是否正确；

2. Check if no communication error: if no error, enter step 3, otherwise do the communication error operation, see section 3.2.10 for details, and then enter step 6;

   检查是否无通信错误：如果是，则进入步骤 3，否则进行通信错误处理-详见 3.2.10 节，然后进入步骤 6；

3. Check status information in the response message: if no error, do the communication recovery operation, see section 3.2.9 for details, otherwise do the communication error operation, see section 3.2.10 for details;

   检查应答帧中的状态信息：如果无错误，则进行通信恢复操作-详见 3.2.9 节，否则进行通信错误处理-详见 3.2.10 节；

4. Update PMIO communication information according to the response status information: set I/O module failure flag, FVActived flag and new i-parameter flag;

   根据应答状态信息更新 PMIO 通信信息：设置 I/O 模块失效标志、故障安全值激活标志和新参数已被使用标志；

5. Update ParOK flag: set with new i-parameter flag;

   更新 ParOK 标志：赋值为 i-parameter 标志的值；

6. Check error flag: if true, set ActiveFV bit in the control byte to 1, otherwise enter step 7;

   检查错误标志位：如果为真，则置控制字节中的 ActiveFV 位为 1，否则进入步骤 7；

7. Check ActiveFV bit: if 1, enter step 8, otherwise end;

   检查 ActiveFV 位：如果为 1，则进入步骤 8，否则结束；

8. Check if auto-recovery: if yes, reset ActiveFV bit in the control byte, otherwise enter step 9;

   检查是否自动恢复：如果是，则复位控制字节中的 ActiveFV 位，否则进入步骤 9；

9. Check if user has confirmed: if yes, reset ActiveFV bit in the control byte, otherwise end.

   检查用户是否已经确认：如果是，则复位控制字节中的 ActiveFV 位，否则结束。

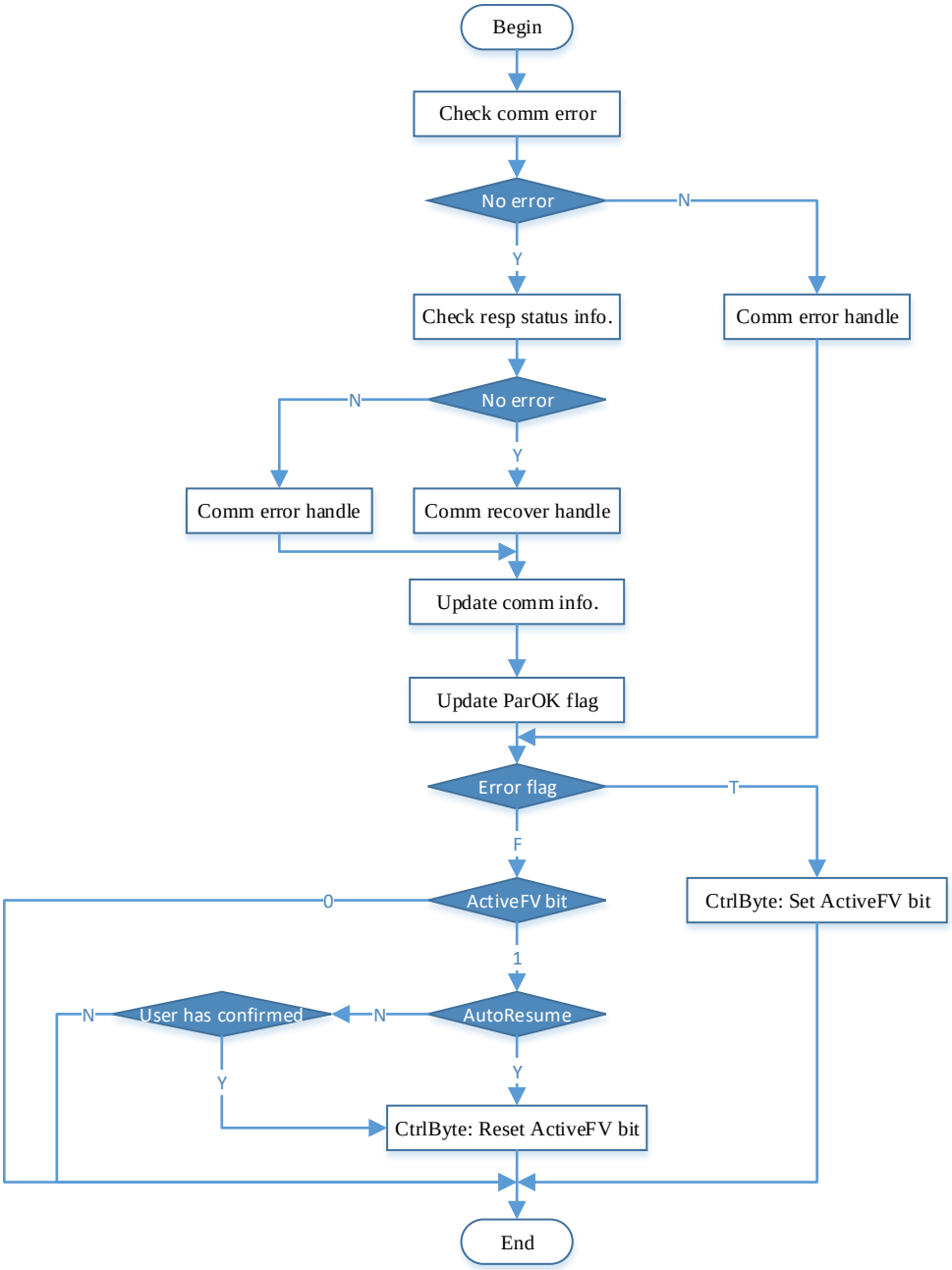

Figure 5-1 decode response message

图 5-1 解析应答帧

5.2.5 Response timeout handling 应答超时处理

SWDD-PM-SP_SafR_SecR_A_020

5.2.5.1 PMIOHandleRecvTO

5.2.5.1.1 Function Description 功能描述

This function is used to handle response timeout.

该函数用于处理应答超时。

5.2.5.1.2 Argument Description 参数说明

- Definition 函数定义

bool_t PMIOHandleRecvTO(uint8_t ucIOID)

- Input argument 输入参数

ucIOID: I/O module ID I/O 模块 ID。

- Output argument 输出参数

If handle successfully.

是否处理成功。

5.2.5.1.3 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. Communication error handling: see section 3.2.10 for details;

   通信错误处理：详见 3.2.10 节；

2. Update control byte: Set ActiveFV flag when error flag is 1;
   Reset ActiveFV flag when error flag is 0.

   更新控制字节：当错误标志为 1 时，置 ActiveFV 标志；
   当错误标志为 0 时，复位 ActiveFV 标志。

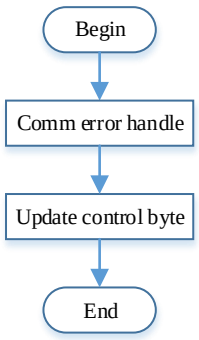

Figure 5-2 handle response timeout
图 5-2 处理应答超时

——以下无正文