Document Title: PM_FW Synchronization module design description of Safety Control System

Document Number: 15-Q04-000076

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: Liu Yang	2015-09-07
Checked by: Zhu Genghua	2016-11-30
Countersigned by: Li Qi, 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	2015-9-7	None	A	Liu Yang	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 同步模块设计说明书

文件编号：15-Q04-000076

项目编号：SF-RD-1501

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：刘　阳	2015 年 9 月 7 日
审核：朱耿华	2016 年 11 月 30 日
会签：李琦　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	2015-9-7		A	刘 阳	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 PM_FW synchronization module 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.

本文档描述安全控制系统 PM_FW 系统状态管理及任务同步模块的设计方案。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。

本文档是软件模块设计的输出，也是后续模块编码的输入。

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

1.2.2 Standard reference documentation 标准参考文档

表 1-1 标准参考文档
Table 1-1 Standard reference documentation

序号	标准号	名称
1	IEC61508: 2010 Part1-7, Ed.2.0	Functional safety of electrical/electronic/programmable electronic safety-related systems
2	IEC61511: 2004, Part1-3	Functional safety - Safety Instrumented Systems for the process industry sector
3	IEC 61131-6:2012	Programmable PMs - Part 6: Functional safety
4	IEC 62443-3-3:2013	Industrial communication networks - Network and system security - Part 3-3: System security requirements and security levels
5	IEC/NP 62443-4-1:2013	Industrial communication networks - Network and system security

		(phase 3)
6	IEC/NP 62443-4-2:2013	IEC/TS 62443-1-3 Ed. 1.0: Security for industrial process measurement and control - Network and system security - Part 1-3: System security compliance metrics
7	ISASecure EDSA 311:2010	ISA Security Compliance Institute – Embedded Device Security Assurance Functional Security Assessment
8	ISASecure EDSA 312:2010	ISA Security Compliance Institute – Embedded Device Security Assurance Software Development Security Assessment
9	IEC 61131-3:2013	Programmable PMs - Part 3: Programming languages

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 synchronization module (marked red) in the software hierarchy is shown below.

同步模块（标红）在软件层次中的位置如下图所示。

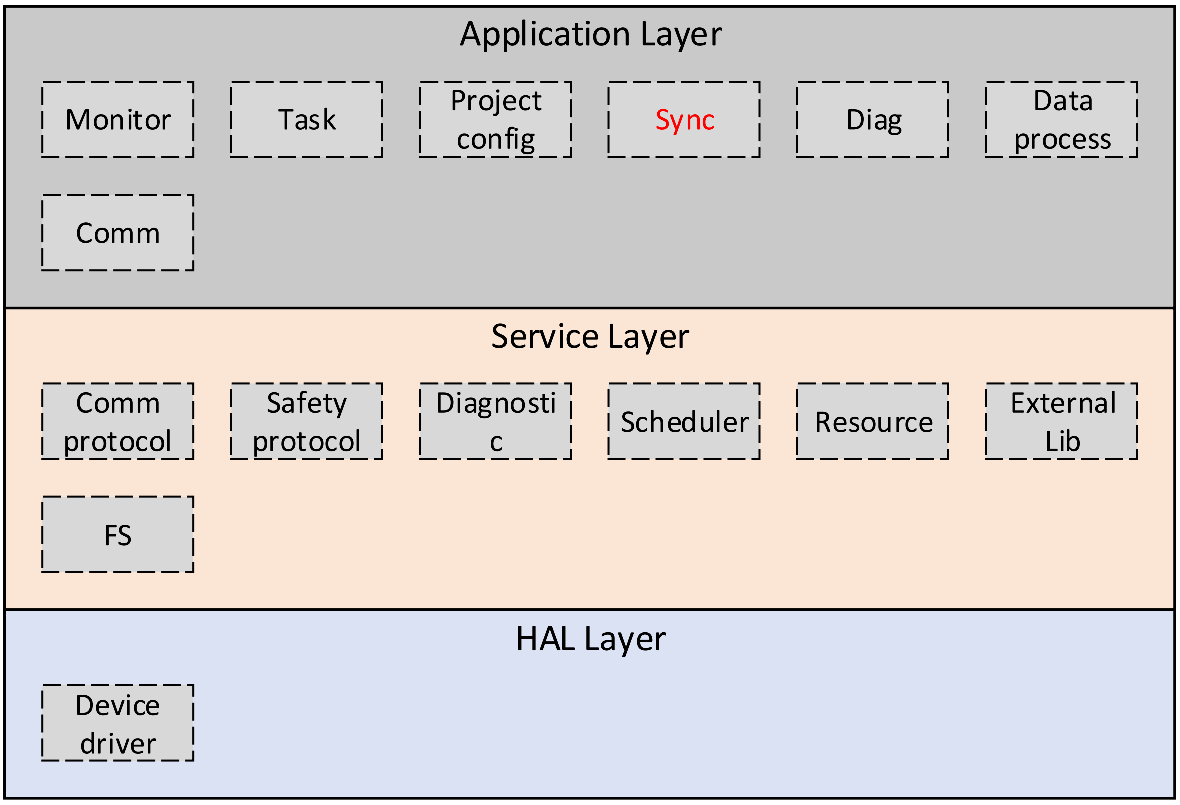

Figure 2-1 the location of the data processing module

图 2-1 模块位置

Synchronization module are mainly used to manage system state and synchronization between PMs.

同步模块主要用于实现对系统三系 PM 的状态管理及同步功能。

3 Module design 模块设计

3.1 Function description 功能描述

The state management implements the state switching and state transition processing between the three PMs of the system, and adds the synchronization task according to the change of the state.

状态管理主要实现系统三系 PM 之间状态交换与状态转换处理等操作，根据状态的变化添加同步任务。

The work that synchronization task needs to complete include:

1. Synchronize the project files, project data, and other system parameters at the system power-on

2. Periodically synchronize PM state;
3. Periodically processing synchronization task, include: internal command and user command.

同步任务管理模块需要完成的工作包括：

1. 上电时同步工程文件、工程数据、系统参数等；
2. 同步 PM 状态；
3. 周期性同步任务，包括内部命令、用户命令等。

The specific steps of synchronization is:

同步的具体步骤为：

1. The state management function adds the state synchronization task to the state synchronization queue;
2. The communication module adds the user command packet to the task synchronization queue, and the state management function adds the internal command packet to the task synchronization queue;
3. The synchronization task takes the state synchronization packet from the state synchronization queue;
4. The synchronization task takes the task synchronization packet from the task synchronization queue;
5. The synchronization task puts the state synchronization packet or the task synchronization packet into the PM_BUS buffer;
6. The synchronization task periodically obtains synchronization packets from the PM_BUS buffer for processing;

1. 状态管理功能将状态同步包添加入同步队列；
2. 通讯模块将用户命令组包添加入任务同步队列，状态管理功能将内部命令包添加入任务同步队列
3. 同步模块从状态同步队列中获取状态同步包
4. 同步模块从任务同步队列中获取任务同步包
5. 同步模块将状态同步包和任务同步包放入链路驱动的缓存中
6. 同步模块周期性的从链路缓存中获取同步包，进行处理

3.2 Design concept 设计思路

3.2.1 System state management 系统状态管理

The management of the system state is based on all states of the three PMs in the system. The possible states of single PM are shown in the following table:

系统状态的管理基于系统内三 PM 的所有状态，单个 PM 可能状态如下表所示：

Table 3-1 State table of single PM

表 3-1 单个 PM 状态表

PM State PM 状态	Description 描述
Offline	PM offline PM 离线
Power-on	PM power-on PM 上电启动
Standby	PM has no project PM 无工程
Stop	Project stop running 工程停止运行
Run	Project is running 工程正在运行

In the above table, Power-on is a transition state, and the other four states are steady state. Each PM will receive the state of the other two PMs, and combine them into a system status code.

上表中 Power-on 为过渡状态，其他 4 种状态为单个 PM 时的稳态。每个 PM 都会接收其他两个 PM 的状态，组合成一个系统状态码。

3.2.2 Power on synchronization 上电同步

The new PM which is power-on is required to synchronize the PM state information and the user data from other PMs. The new PM first enter the power-on ready phase after power-on, then enter the power-on synchronization phase, and finally into the periodic synchronization.

新 PM 上电需要进行 PM 状态信息的同步和用户数据的同步。新 PM 上电后，先进入上电准备状态，然后进入上电同步状态，最后进入周期性同步。

3.2.2.1 Power-on ready phase 上电准备阶段

When all initialization preparations are completed, the PM broadcasts power-on request to other online PMs, and then waiting for response from other PMs. If no response is received, it remains in this state.

当所有初始化准备工作做好后，处于上电准备阶段的 PM 广播上电请求，等待其他在线 PM 给予应答，若接收不到应答，则一直处于此状态。

When the PM is in the power-on synchronization phase, it does not respond to power-on requests from other PMs.

当 PM 处于上电同步状态时，不响应其他系 PM 的上电请求。

3.2.2.2 Power-on synchronization phase 上电同步阶段

The PM in power-on synchronization phase needs to complete user data synchronization and state changes, the detailed processing steps are as follows:

处于上电同步状态的 PM 要完成用户数据的同步和状态的转变，具体处理步骤如下：

1. The PM in power-on synchronization phase synchronizes its own state to other on-line PM.

1．处于上电同步状态的 PM 同步自身状态给其他在线 PM。

2. Project files judgement: the power-on PM compares with other online PMs to judge whether the project files are valid and whether it needs to be cleared.

2．工程文件判断：上电 PM 通过与系统内其他在线 PM 比较，判断自身工程文件是否有效，是否需要清除。

3. Synchronize the project files: According to the state of the project, determine whether to add synchronize project file tasks to the synchronization task queue.

3．同步工程文件：根据本机工程状态，判断是否添加同步工程文件任务到同步任务队列。

4. Complete synchronize project file task.

4．完成同步工程文件任务。

5. Synchronize real-time data: After synchronizing the project files, add the synchronize real-time data task to the synchronization queue. When the system is in the Run state，The PM in the power-on synchronization phase notifies other PMs to pause the UP.

5．同步实时数据：完成同步工程文件后，添加同步实时数据任务。如果系统处于 RUN 状态，同步实时数据前，上电同步状态的 PM 通知其他在 Run 状态的 PM 暂停运行 UP。

6. Complete synchronize real time data task.

6．完成同步实时数据任务。

7. PM state transition: According to the system state, complete the PM state transition.

7．PM 状态转变：根据系统状态，完成 PM 状态转变。

Specific flow chart is as follows:

具体流程图如下：

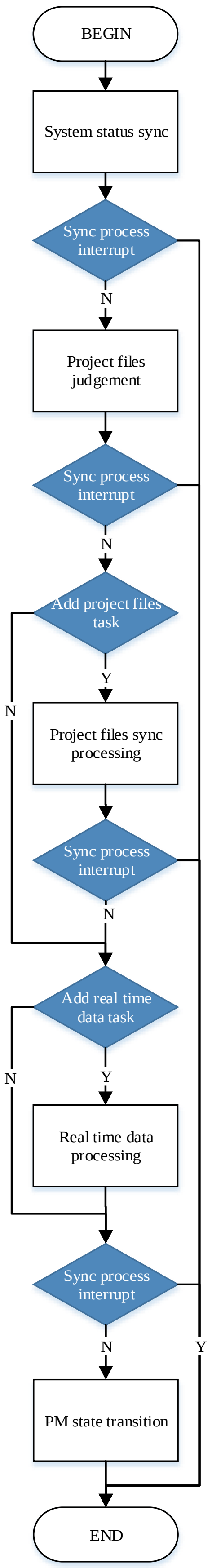

Figure 3-1 Power on sync processing

图 3-1 上电同步处理

3.2.2.3 PM State Transition PM 状态转变

When the user data synchronization is complete, if the system is in Run state, the power on PM sends an internal Run command to other PMs, and if the system is in Stop state, the power on PM automatically switches into the Stop state.

用户数据同步完成后，若系统处于 Run 状态，完成上电数据同步的 PM 发送同时进入 Run 的状态的内部命令；若系统处于 Stop 状态，完成上电数据同步的 PM 自动转入 Stop 状态。

The processing flow of RUN state is as follows.

Run 状态时的处理流程如下。

1. Parse the project file, complete the parameter configuration.
2. Calculates the start time of the synchronization run.
3. Send the internal synchronization run command.
4. Waiting for other online PM’s reply and start the UP at the same time.
5. Set the synchronization status to periodic synchronization.

   1. 解析工程文件，完成参数配置；
   2. 计算同步运行起始时间；
   3. 发送内部同步运行命令；
   4. 等待接收其他在线暂停运行的 PM 回复应答，接收到所有应答，按照所定时刻，同时开始运行 UP；超时未接收所有应答。重新发送同步运行命令；
   5. 置同步状态为周期同步。

The processing flow of Stop state is as follows.

Stop 状态时的处理流程如下。

1. Parse the project file, complete the parameter configuration;
2. The PM set itself state as Stop;
3. Set the synchronization status to periodic synchronization.

   1. 解析工程文件，完成参数配置；
   2. PM 设置本身状态为 Stop；
   3. 置同步状态为周期同步。

The processing flow of Standby state is as follows.

Standby 状态时的处理流程如下。

1. No configuration required, PM set itself state as Standby;
2. Set the synchronization status to periodic synchronization.

1. 无需配置，PM 设置本身状态为 Standby；
2. 置同步状态为周期同步。

3.2.3 Synchronization task 同步任务

The synchronization module periodically processes the synchronization packets in the synchronization state queue and the synchronization task queue, sends the synchronization packets to other PMs, periodically receives the synchronization packets from other PMs, and process them.

同步模块周期性的处理同步状态队列和同步任务队列中的同步包，将同步包发送给其他 PM；周期性接收其他 PM 发送的同步包，并进行处理。

In each period, synchronization module processes only one task synchronization packet and one state synchronization packet.

每周期，同步模块只处理一次任务同步包和一次状态包同步。

3.2.4 Synchronize Queue同步队列

Each PM maintains two synchronization queues: the task synchronization queue and the state synchronization queue. The task synchronization queue is used by each module to add synchronization tasks to the queue for synchronization. The state synchronization queue is used to synchronize the state information of PMs.

每一系 PM 维护两个同步队列：任务同步队列和状态同步队列。任务同步列用于各模块同步任务添加到任务同步队列进行同步；状态同步队列用处同步 PM 的状态信息。

The task synchronization queue is a first-in, first-out, round-robin queue. In each PM_FW execution cycle (5ms) at most one task is processed. The queue size is designed as 20. When the queue is full, no more tasks can be added to the queue.

任务同步队列为先入先出的循环队列，每个 PM_FW 执行的周期（5ms）最多只处理一个任务。队列大小设计为 20。当队列中满时，不再向队列中添加任务。

The task synchronization queue is a first-in, first-out, round-robin queue. In each PM_FW execution cycle (5ms) at most one task is processed. The queue size is designed as 5. When the queue is full, no more tasks can be added to the queue.

状态同步队列为先入先出的循环队列，每个 PM_FW 执行的周期（5ms）最多只处理一个任务。队列大小设计为 5。当队列中满时，不再向队列中添加任务。

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 SyncInit(void)

Input argument 输入参数	Output argument 输出参数	Description 描述
No. 无。	No. 无。	Synchronization initialization function 同步初始化函数

2. void SyncMngt(void)

Input argument 输入参数	Output argument 输出参数	Description 描述
No. 无。	No. 无。	Synchronization management function 同步管理模块

3. void StateMngtInit(void)

Input argument 输入参数	Output argument 输出参数	Description 描述
No. 无。	No. 无。	State management initialization 状态管理初始化处理

4. void SysStateMngt(void)

Input argument 输入参数	Output argument 输出参数	Description 描述
No. 无。	No. 无。	State management function 状态管理函数

3.4 Global variable 全局变量

Table 3-2 Global variable list

表 3-2 全局变量列表

变量名称	变量描述
static SyncFileStatus_t s_stSyncFileStatus;	The state of the file during synchronization 同步时文件的状态
static SyncDataStatus_t s_stSyncDataStatus;	The state of the data during synchronization 同步时数据的状态
static SyncTimerStatus_t	The time when synchronization happened

s_stSyncTimer;	同步发生时间
static SyncData_t s_stSyncData;	The data which will be synchronized 需要被同步的数据
static File_Handle_Flag_t emFileHandleFlg;	It is used to record the current state of the file 记录文件的当前处理状态
static uint32_t s_uiUnstableCode [MAX_UNSTABLE_STATE]	Unstable code table 非稳态状态码表
work_status_t emlastPreWorkStat	The previous PM work state 前一系 PM 工作状态
work_status_t emlastNextWorkStat	The next PM work state 后一系 PM 工作状态

3.5 Data structure 数据结构

```
typedef enum
{
    NEW_FILE = (0x00000000U);
    NOT_NEW_FILE;
}file_send_type_t;

typedef enum
{
    AREA_G_ID = 0x00000000U;
    AREA_Q_ID;
    AREA_I_ID;
    MAX_AREA_NUM;
}data_areaid_t;

typedef enum
{
    SRV_DATA = 0x00000000U;
    SRVACK_DATA;
    CTRL_DATA;
}srv_data_type_t;

typedef enum
{
    SYNC_FILE = (0x00000000U);
```

```
        SYNC_REAL_DATA;
        SYNC_COMPLETE;
}sync_srv_type_t;

typedef enum
{
        LAST_PACK = (0x00000000U);
        NOT_LAST_PACK;
}pack_type_t;

typedef enum
{
        S_DATA_NULL = 0x00000000U;
        S_SRV_DATA;
        S_SRVACK_DATA;
        S_PRG_DATA;
}sync_data_type_t;

typedef struct SyncFileStatusTag
{
        SysCfgFileType_t emCfgFileType;
        uint32_t uiFileSize;
        uint32_t uiSendSize;
        file_send_type_t emStartFlag;
}SyncFileStatus_t;

typedef struct SyncDataStatusTag
{
        Task_ID_t emTaskID;
        SysRtDataType_t emAreaId;
        uint32_t uiCurOffset;
        uint32_t uiCurDataLen;
}SyncDataStatus_t;

typedef struct SyncParamStatusTag
{
        uint32_t uiCurOffset;
```

```
}SyncParamStatus_t;

typedef struct SyncTimerStatusTag
{
    SyncTime64_t unTaskStartTime;
}SyncTimerStatus_t;

typedef struct SyncOneSyncDataTag
{
    srv_data_type_t emType;
    srv_srvdata_type_t emSubType;
    ack_flag_t emAckFlag;
    uint8_t ucData[CFG_MAX_SYNCDATA_LEN];
}SyncOneSyncData_t;

typedef struct SyncCtrlSrvTag
{
    uint32_t uiSrvId;
    srv_srvdata_type_t emSrvType;
    SyncTime64_t unActiveTime;
    SyncTime64_t unTimeOutTime;
}SyncCtrlSrv_t;

typedef struct SyncHeadTag
{
    sync_srv_type_t emSrvType;
}SyncHead_t;

typedef struct SyncFileHeadTag
{
    SysCfgFileType_t emCfgFileType;
    pack_type_t emLastPackFlag;
    uint32_t uiFileOffset;
}SyncFileHead_t;

typedef struct SyncFileTag
{
    SyncHead_t stSyncHead;
    SyncFileHead_t stFileHead;
    uint8_t ucData[CFG_MAX_SYNC_FILE_SIZE];
}SyncFile_t;

typedef struct SyncDataHeadTag
{
    Task_ID_t emTaskID;
    SysRtDataType_t emAreaId;
    uint32_t uiOffset;
    uint32_t uiDataLen;
    pack_type_t emLastPackFlag;
}SyncDataHead_t;

typedef struct SyncRealDataTag
{
    SyncHead_t stSyncHead;
    SyncDataHead_t stDataHead;
    uint8_t ucData[CFG_MAX_SYNC_DATA_SIZE];
}SyncRealData_t;

typedef struct SyncParamHeadTag
{
    uint32_t uiOffset;
    pack_type_t emLastPackFlag;
}SyncParamHead_t;

typedef struct SyncIoParamTag
{
    SyncHead_t stSyncHead;
    SyncParamHead_t stParamHead;
    uint8_t ucData[CFG_MAX_SYNC_DATA_SIZE];
}SyncIoParam_t;

typedef struct SyncStaticAckTag
{
    SyncHead_t stSyncHead;
    uint8_t ucData[CFG_MAX_STATIC_ACK_SIZE];
}SyncStaticAck_t;

typedef struct SyncSrvDataTag
{
    SysPMStateInfo_t stPMState;
    uint32_t uiSyncSrvDataLen;
    uint8_t ucSrvData[CFG_MAX_SYNC_SRV_SIZE];
}SyncSrvData_t;

typedef struct SyncPrjDataTag
{
    uint32_t uiSyncPrjDataLen;
    uint8_t ucPrjData[MAX_PRJ_SYNC_SIZE];
}SyncPrjData_t;

typedef struct SyncMSGTag
{
    SyncSrvData_t stSyncSrvData;
    SyncPrjData_t stSyncPrjData;
}SyncMSG_t;

typedef struct SyncDataTag
{
    int16_t sLinkASrvLen;
    int16_t sLinkBSrvLen;
    int16_t sLinkAPrjLen;
    int16_t sLinkBPrjLen;
    int16_t sLinkASrvErr;
    int16_t sLinkBSrvErr;
    int16_t sLinkAPrjErr;
    int16_t sLinkBPrjErr;
    SyncMSG_t stSyncRecvMSGA;
    SyncMSG_t stSyncRecvMSGB;
    SyncMSG_t stSyncSendMSGA;
    SyncMSG_t stSyncSendMSGB;
}SyncData_t;
```

```
typedef enum
{
    WS_OFFLINE = 0x00000000U;
    WS_STANDBY;
    WS_WORK;
    MAX_WORK_STATUS;
    WS_INVALID_TYPE;
}work_status_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 3-4 sub function list

表 3-4 子功能列表

Sub function No. 子功能编号	Function description 功能描述
SWDD-PM-SYNC_SafR_NSecR_A_001	Synchronization management initialization 同步管理初始化
SWDD-PM-SYNC_SafR_NSecR_A_002	Synchronization management runs periodically 同步管理周期运行
SWDD-PM-SYNC_SafR_NSecR_A_003	State management initialization 状态管理初始化
SWDD-PM-SYNC_SafR_NSecR_A_004	State management runs periodically 状态管理周期运行

4 Design of sub-function 子功能设计

4.1 Synchronization management initialization 同步任务初始化

SWDD-PM-SYNC_SafR_NSecR_A_001

4.1.1 SyncInit

4.1.1.1 Function Description 功能描述

This function completes initialization of module.

该函数完成模块的初始化。

4.1.1.2 Argument Description 参数说明

- Definition 函数定义

void SyncInit(void)

- Input argument 输入参数

No.

无。

- Output argument 输出参数

No.

无。

4.1.1.3 Processing flow 处理流程

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

此函数无分支，流程图省略。

4.2 Synchronization task management function 同步任务管理函数

SWDD-PM-SYNC_SafR_NSecR_A_002

4.2.1 SyncMngt

4.2.1.1 Function Description 功能描述

This function completes the management and execution of the synchronization task, reception and processing of the synchronized data. Includes the synchronization and processing of the status data.

该函数完成同步任务的管理执行、同步数据的接收及处理。包含状态数据的同步和处理。

4.2.1.2 Argument Description 参数说明

- Definition 函数定义

void SyncMngt(void)

- Input argument 输入参数

No.

无。

- Output argument 输出参数

No.

无。

4.2.1.3 Processing flow 处理流程

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

流程如下图所示，主要步骤如下：

1. Receive the status data transmitted from the other two PMs;

   接收从其它两系 PM 传送过来的状态数据；

2. If the received data are error, do the error data processing, otherwise process the received data;

   若接收数据出现错误，则进行错误数据处理，否则处理接收数据；

3. Check if need to initialize the synchronization information record: if yes, initialize it and its corresponding information;

   判断是否需初始化同步信息记录：如果是，则初始化记录及其相关信息；

4. Check if there are synchronization tasks need to be processed: if none, initialize the synchronization information record;

   判断是否有同步任务需要处理：若无，则初始化同步信息记录；

5. Get the first synchronization task if the queue is not empty;

   若同步任务队列不为空，获取第一个同步任务；

6. Check if need to wait, if needed, wait, otherwise process it.

   判断是否需等待，若需要，则等待，否则处理此同步任务。

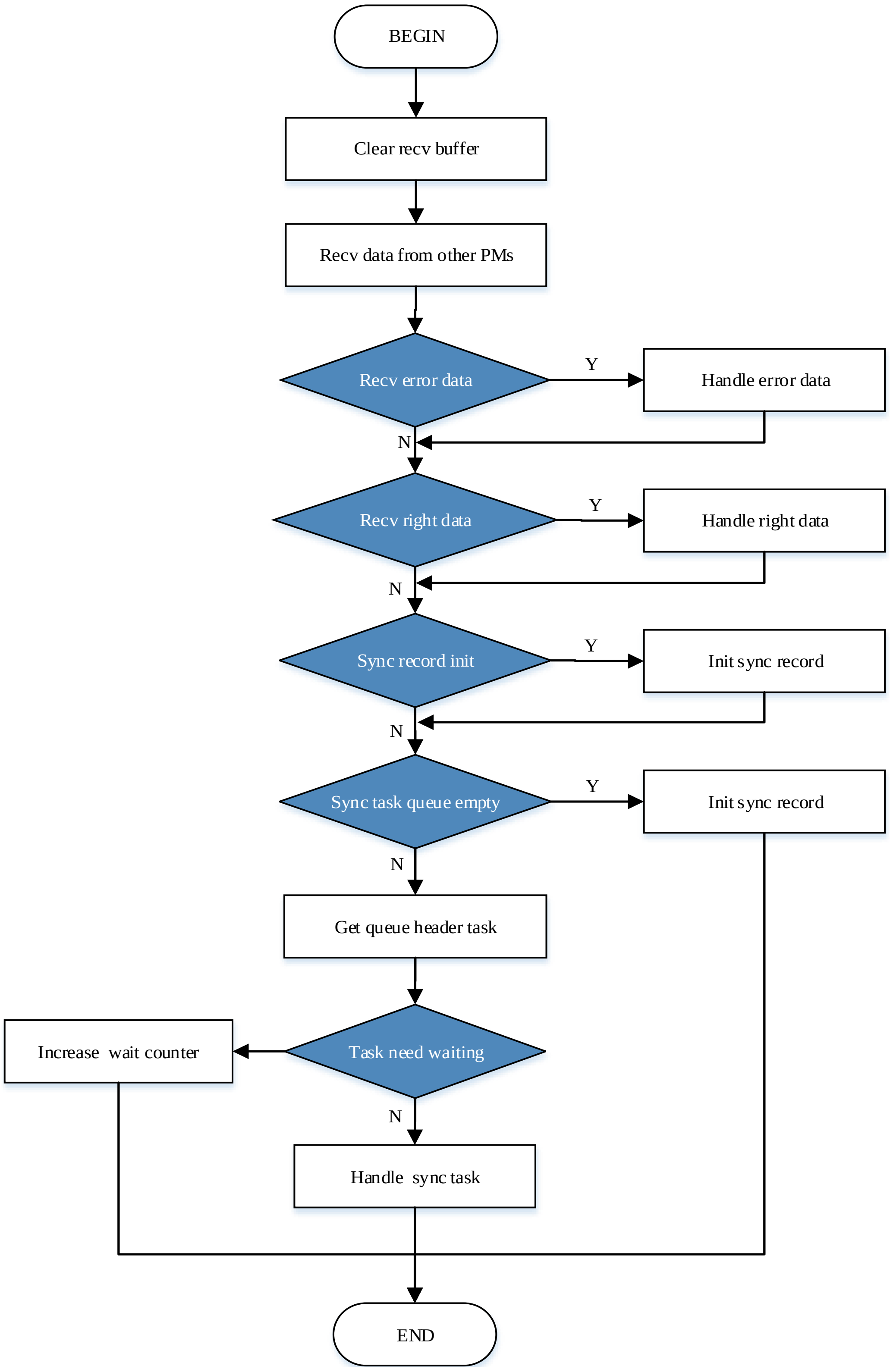

Figure 4-1 sync task management process
图 4-1 同步任务管理处理流程

4.3 System state management module initialization 系统状态管理模块初始化

SWDD-PM-SYNC_SafR_NSecR_A_003

4.3.1 StateMngtInit

4.3.1.1 Function Description 功能描述

This function completes initialization of module.

该函数完成模块的初始化。

4.3.1.2 Argument Description 参数说明

- Definition 函数定义

void StateMngtInit(void)

- Input argument 输入参数

No.

无。

- Output argument 输出参数

No.

无。

4.3.1.3 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。

4.4 System state management function 系统状态管理函数

SWDD-PM-SYNC_SafR_NSecR_A_004

4.4.1 SysStateMngt

4.4.1.1 Function Description 功能描述

This function completes the system periodic state management work.

该函数完成系统周期性的状态管理工作。

4.4.1.2 Argument Description 参数说明

- Definition 函数定义

void SysStateMngt(void)

- Input argument 输入参数

No.

无。

- Output argument 输出参数

No.

无。

4.4.1.3 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. Get status of all three PMs;

   获取三系 PM 的状态；

2. Check if local status change: if yes, broadcast it;

   判断本机状态是否变化：如果是，则广播本机状态；

3. Generate system state code according to the status of all three PMs, if changed, enter step 4;

   根据三系 PM 的状态生成系统状态码，如果变化，则进入步骤 4；

4. Check if non-steady state, if yes, process this non-steady state;

   判断是否为非稳态，如果是，则进行非稳态处理；

5. Check if steady state, if yes, process this steady state.

   判断是否为稳态，如果是，则进行稳态处理。

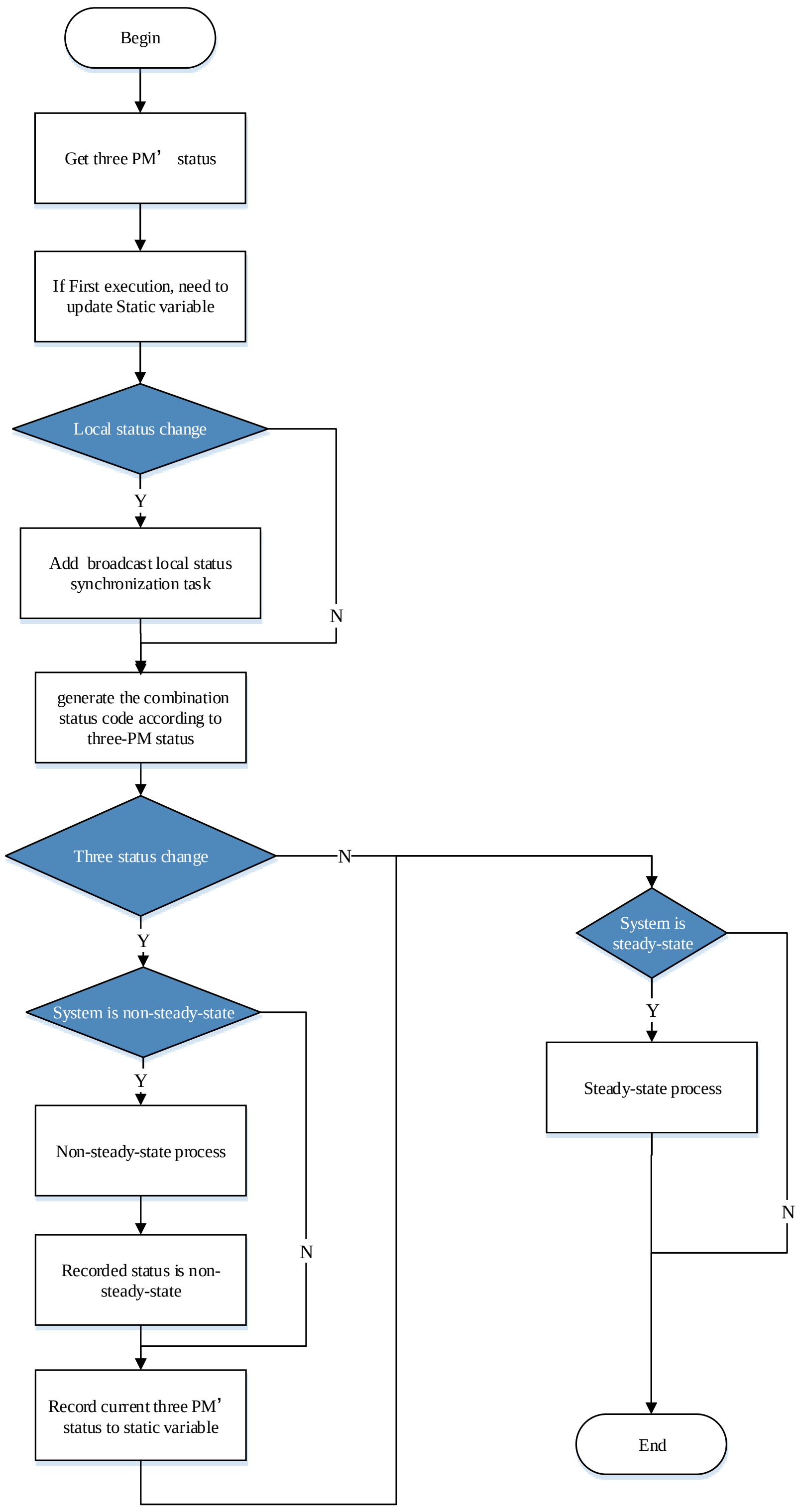
Begin
Get three PM' status
If First execution, need to update Static variable
Local status change
Y
N
Add broadcast local status synchronization task
generate the combination status code according to three-PM status
Three status change
N
Y
System is non-steady-state
Y
N
Non-steady-state process
Recorded status is non-steady-state
Record current three PM' status to static variable
System is steady-state
Y
N
Steady-state process
End

Figure 4-2 system state management process

图 4-2 系统状态管理处理流程

——以下无正文