Document Title: CM_FW device driver module design description of Safety Control System

Document Number: 17-Q04-000534

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 Yue 2016-12-31

Checked by: Zhu Genghua 2017-1-10

Countersigned by: Li Qi, Wang Dong

Approved by: Wen Yiming 2017-1-10

Revision History

No.	Relevant Chapter	Change Description	Date	Version Before Change	Version After Change	Prepared by	Checked by	Approved by
1		Document created	2016-12-31	None	A	Liu Yue	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 设备驱动模块设计说明书

文件编号：17-Q04-000534

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：刘跃 2016 年 12 月 31 日

审核：朱耿华 2017 年 1 月 10 日

会签：张磊

批准：温宜明 2017 年 1 月 10 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	2016-12-31		A	刘跃	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 device driver 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 设备驱动模块的设计方案。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 device driver module (marked red) in the software hierarchy is shown below.

设备驱动模块（标红）在软件层次中的位置如下图所示。

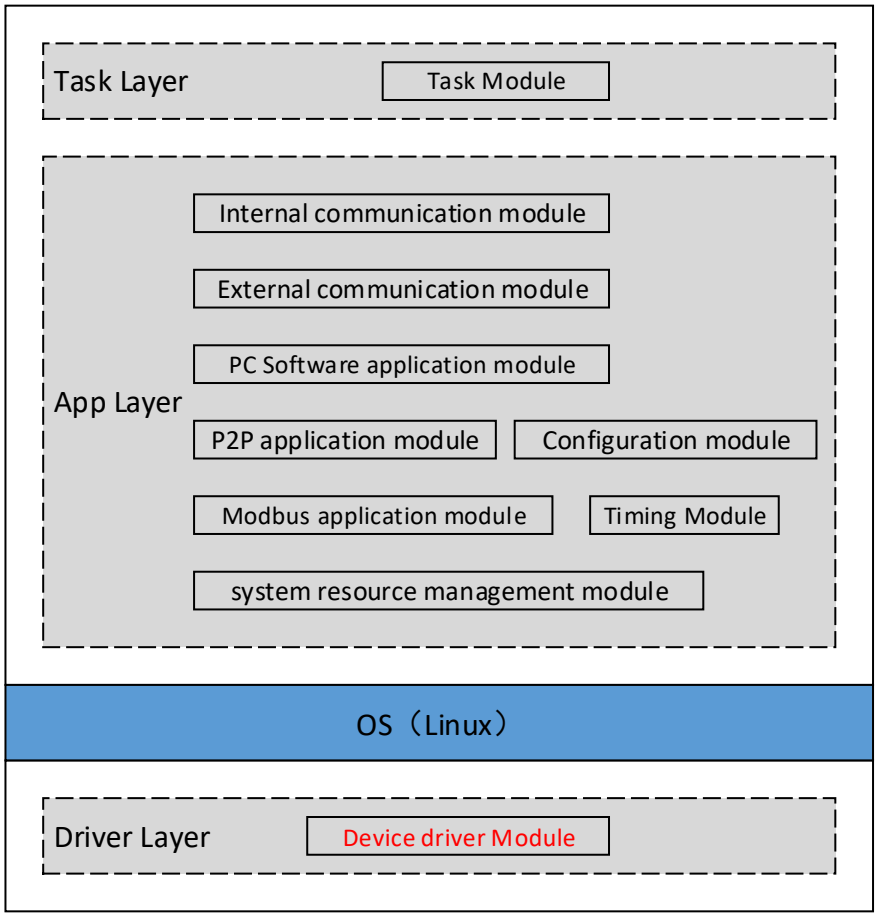

Figure 2-1 the location of the data processing module

图 2-1 模块位置

Device driver module is used to operate various device, includes CM_BUS, FPGA, network, etc.

设备驱动模块主要用于操作各种设备，包括 CM_BUS、FPGA、网络等。

3 Module design 模块设计

3.1 Function description 功能描述

This document describes four parts of the peripheral driver: the reception and transmission of PCI-E data, the register operation with the FPGA, the read/write of SPI controller and the network storm protection function.

本文档描述了外设驱动的四个部分， PCI-E 数据的接收与发送，与 FPGA 的寄存器操作，SPI 控制器的读写，防网络风暴功能。

The driver makes a specific part of the hardware response to a defined internal programming interface that completely hides the details of the device's activities. The user's activities are carried out through a set of standardized calls that are independent of the particular driver. The aim is to map these calls to the actual hardware and device-related operations. The driver can separate the kernel with user program. It is a “plug & play” module and can be loaded when necessary.

驱动程序依据硬件定义了一套内部接口，此内部接口隐藏了硬件的内部细节，使得用户程序可以使用一套标准的 API 接口来调用驱动程序的功能函数。驱动程序的作用就是将用户程序的 API 映射到实际的硬件设备上来完成相应的操作。驱动程序是将内核与用户程序隔离开来。驱动程序是即插即用模块，在需要时候加载即可。

3.2 Design concept 设计思路

3.2.1 CM_BUS driver CM_BUS驱动

The CM_BUS module receive and send data through the PCIE bus, and the process is as follows:

1. Initialize and register the CM_BUS driver.
2. Call the Linux ioremap function to map the address area of the FPGA.
3. Read and write access to the FPGA's RAM area to receive and send data.

CM_BUS 是控制器通过 PCIE 总线接收和发送数据，处理流程如下：

1. 初始化与注册 CM_BUS 驱动程序。
2. 调用 Linux ioremap 函数映射 FPGA 的地址区域。
3. 通过读写访问 FPGA 的 RAM 区来接收和发送数据。

3.2.2 FPGA register read and write FPGA寄存器读写

Read and write FPGA register flow is as follows:

1. Initialize and register the FPGA driver.
2. Call the Linux ioremap function to map the FPGA's REG address area.

3. Read and write access to the FPGA's REG to operate the register.

读写 FPGA 的寄存器流程如下：

1. 初始化与注册 FPGA 驱动程序。
2. 调用 Linux ioremap 函数映射 FPGA 的 REG 地址区域。
3. 通过读写访问 FPGA 的 REG 来操作寄存器。

3.2.3 SPI controller read/write SPI控制器读写

Read and write SPI controller flow is as follows:

1. Configure the GPIO and SPI controller.
2. Initialize and register the SPI driver.
3. Invoke the Linux ioremap function to map the mapped registers of the SPI controller.
4. Read and write access to the SPI registers to send and receive data.

读写 SPI 的流程如下：

1. 配置 GPIO 以及 SPI 控制器
2. 初始化与注册 SPI 驱动程序.
3. 调用 Linux ioremap 函数映射 SPI 控制器的 mapped registers.
4. 通过读写访问 SPI 的 registers 发送和接收数据.

3.2.4 CM network storm protection CM防网络风暴

By judging the number of data packets to determine whether a network storm occurred. When the network storm occurs, the following steps will be executed:

1. Close the network card.
2. The application does not send packets.
3. Periodically open the network card to detect whether the network storm occurs. If the storm is not over, the network card will close again.

通过判断 CM 网卡的数据包流量来判断是否发生了网络风暴，当发生网络风暴时下列步骤将被执行：

1. 关闭网卡接收数据功能.
2. 应用程序不发送数据包.
3. 周期性地打开网卡进行探测，如果网络风暴还未结束就再次关闭网卡。

3.2.5 CM Linux configuration CM Linux系统配置

SWDD-CM-DD_NSafR_SecR_A_006

By cutting the Linux configuration file, you can remove some of the extra peripheral drivers and system functions, the system does not support peripheral functions are as follows:

1. Sound Device Driver.
2. eSDHC Driver.
3. SATA driver.
4. USB driver.
5. Graphics driver.
6. I2C device driver.

通过裁剪 Linux 配置文件，可以删除一些多余的外设驱动以及系统功能，系统不支持的外设功能如下：

1. Sound Device Driver.
2. eSDHC Driver.
3. SATA driver.
4. USB driver.
5. Graphics driver.
6. I2C device driver.

The root file system is the jffs2 file format, and storage device is Nor Flash. The modified Nor Flash partition structure is shown as follows:

我们使用 jffs2 格式的根文件系统，存储设备是 Nor Flash，修改后的 Nor Flash 分区结构如下图：

Start offset	End offset	Description	Size
0x00000000	0x0001FFFF	DTB image	128K
0x00020000	0x005FFFFF	Kernel image	5M
0x00600000	0x03dbFFFF	JFFS2 file system	55M
0x03ec0000	0x03FFFFFF	Boot loader	1M

By default, Linux provides many kinds of network services: SSH (port 22), FTP (port 20, 21), TFTP

(port 69) and Telnet (port 23). For information security, these services are disabled. The NTP service is enabled and only port 123 is used to send request packets to the NTP server, and a temporary port (port number larger than 30000) is used to receive the reply packets sent from the server.

Linux 系统默认提供了多种网络服务： SSH（port 22）、FTP（port 20,21）、TFTP（port 69）、Telnet（port 23），为了信息安全考虑这些服务均被关闭，对应的端口不再使用。 同时打开了 NTP 服务，只使用 123 端口向 NTP 服务器发送请求报文，并且会使用一个临时端口（30000 以上的端口）接收服务器发送的应答报文。

3.3 Interface function 接口函数

3.3.1 CM_BUS driver CM_BUS驱动

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. static long pcie_ioctl(struct file *fp, unsigned int send, unsigned int recv)

Input argument 输入参数	Output argument 输出参数	Description 描述
struct file *fp, unsigned int send, unsigned int recv fp:文件指针，send：写人寄存器的值，recv：读入寄存器的值	No. 无。	Read and write registers 读写寄存器

2. static ssize_t pcie_send(struct file *fp, const char __user *user_buffer, size_t count, uint32_t cmd)

Input argument 输入参数	Output argument 输出参数	Description 描述
struct file *fp, const char __user *user_buffer, size_t count, uint32_t cmd Fp:文件指针，user_buffer：用户buffer，cmd：命令字	No. 无。	Send data interface function. 发送数据处理调用接口

3. static ssize_t pcie_recv(struct file *fp, const char __user *user_buffer, size_t count,

uint32_t cmd)

Input argument 接口输入参数	Output argument 接口输出参数	Description 描述
struct file *fp, const char __user *user_buffer, size_t count, uint32_t cmd Fp:文件指针，user_buffer：用户buffer，cmd：命令字	No. 无。	Receive data interface function. 接收数据处理调用接口

3.3.2 FPGA register read and write FPGA寄存器读写

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. static int localbus_init(void)

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

2. static int localbus_ioctl (struct file *file, unsigned int cmd, unsigned long arg)

Input argument 输入参数	Output argument 输出参数	Description 描述
struct file *file, unsigned int cmd, unsigned long arg Fp:文件指针，cmd：命令字，arg：参数	No. 无。	Read and write registers 读写寄存器

3.3.3 SPI controller read/write SPI控制器读写

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. static int spi_init(void)

Input argument	Output argument	Description

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

2. static ssize_t spi_write_data(struct file *fp,const char __user *user_buffer, size_t count, uint32_t cmd)

Input argument 输入参数	Output argument 输出参数	Description 描述
struct file *fp,const char __user *user_buffer, size_t count, uint32_t cmd Fp:文件指针，user_buffer：用户buffer，count：写入的长度，cmd：命令字	No. 无。	Write data interface function. 写数据调用接口

3.3.4 CM network storm protection CM防网络风暴

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. unsigned int get_eth1(void)

Input argument 输入参数	Output argument 输出参数	Description 描述
No. 无。v	No. 无。	Get the network card 1 status 获取网卡 1 状态

2. unsigned int get_eth2(void)

Input argument 输入参数	Output argument 输出参数	Description 描述
No. 无。	No. 无。	Get the network card 2 status 获取网卡 2 状态

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.	uint32_t *	ioremap_fpgaram_addr	FPGA RAM address FPGA RAM 地址
2.	uint32_t	speedInterval	Threshold 门限值
3.	uint32_t	restartTime	Network card open time interval 网卡间隔打开时间

3.5 Data structure 数据结构

1. File operation structure

```
static const struct file_operations pcie_xfer_fops = {
    .owner = THIS_MODULE,
    .read    = pcie_recv,
    .write = pcie_send,
    .unlocked_ioctl = pcie_ioctl,
     };
```

2. Device structure

```
static struct miscdevice pcie_xfer_device = {
    .minor = MISC_DYNAMIC_MINOR,
    .name = "pcie_xfer",
    .fops = &pcie_xfer_fops,
};
```

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-DD_NSafR_NSecR_A_001	CM_BUS data input/output. CM_BUS data 输入和输出

SWDD-CM-DD_NSafR_NSecR_A_002	FPGA register read/write FPGA 寄存器读写
SWDD-CM-DD_NSafR_NSecR_A_003	SPI controller read/write. SPI 控制器读写
SWDD-CM-DD_NSafR_SecR_A_004	Network storm protection 网络风暴防护
SWDD-CM-DD_NSafR_NSecR_A_005	SRAM read/write 读写 SRAM

4 Design of sub-function 子功能设计

4.1 CM_BUS driver CM_BUS 驱动

SWDD-CM-DD_NSafR_NSecR_A_001

4.1.1 pcie_recv

4.1.1.1 Function Description 功能描述

This function is used to process the CM_BUS receive data cyclically.

本函数用于进行 CM_BUS 数据接收处理。

4.1.1.2 Argument Description 参数说明

- Function Definition 函数定义

static ssize_t pcie_recv(struct file *fp,const char __user *user_buffer, size_t count, uint32_t cmd)

- Input argument 输入参数

Fp: file pointer, 文件指针

user_buffer: user buffer, 用户 buffer

count: data length, 写入的长度

cmd: command word, 命令字

- Output argument 输出参数

Returns the number of bytes received

返回接收的字节数

4.1.1.3 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。

4.1.2 pcie_send

4.1.2.1 Function Description 功能描述

This function is used to process the CM_BUS send data cyclically.

本函数用于进行 CM_BUS 数据发送处理。

4.1.2.2 Argument Description 参数说明

- Function Definition 函数定义

static ssize_t pcie_send(struct file *fp,const char __user *user_buffer, size_t count, uint32_t cmd)

- Input argument 输入参数

Fp: file pointer, 文件指针

user_buffer: user buffer, 用户 buffer

count: data length, 写入的长度

cmd: command word, 命令字

- Output argument 输出参数

Returns the number of bytes sent

返回发送成功的字节数

4.1.2.3 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 FPGA register read and write FPGA 寄存器读写

SWDD-CM-DD_NSafR_NSecR_A_002

4.2.1 fpga_ioctl

SWDD-CM-DD_NSafR_NSecR_A_003

4.2.1.1 Function Description 功能描述

This function is used to read and write FPGA registers.

本函数用于进行 FPGA 寄存器的读写操作。

4.2.1.2 Argument Description 参数说明

- Function Definition 函数定义

static int fpga_ioctl (struct file *file, unsigned int cmd, unsigned long arg)

- Input argument 输入参数

file: file pointer, 文件指针

cmd: command word, 命令字

arg: configuration parameter, 配置参数

- Output argument 输出参数

Returns 0 for successful operation.

Return negative for failed operation.

成功返回 0。

失败返回负数。

4.2.1.3 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。

4.3 SPI write 写 SPI

4.3.1 spi_write_data

SWDD-PM-CMDD_NSafR_NSecR_A_003

4.3.1.1 Function Description 功能描述

This function is used to send data via SPI.

本函数用于进行通过 SPI 发送数据。

4.3.1.2 Argument Description 参数说明

- Function Definition 函数定义

static ssize_t spi_write_data(struct file *fp, const char __user *user_buffer, size_t count, uint32_t cmd);

- Input argument 输入参数

fp : File pointer. 文件指针

user_buffer : User data buffer. 用户数据缓存。

count : The number of bytes to write. 要写的字节数。

cmd : Operation command. 操作命令。

- Output argument 输出参数

Return OK for successful operation.

操作成功返回 OK。

4.3.1.3 处理流程

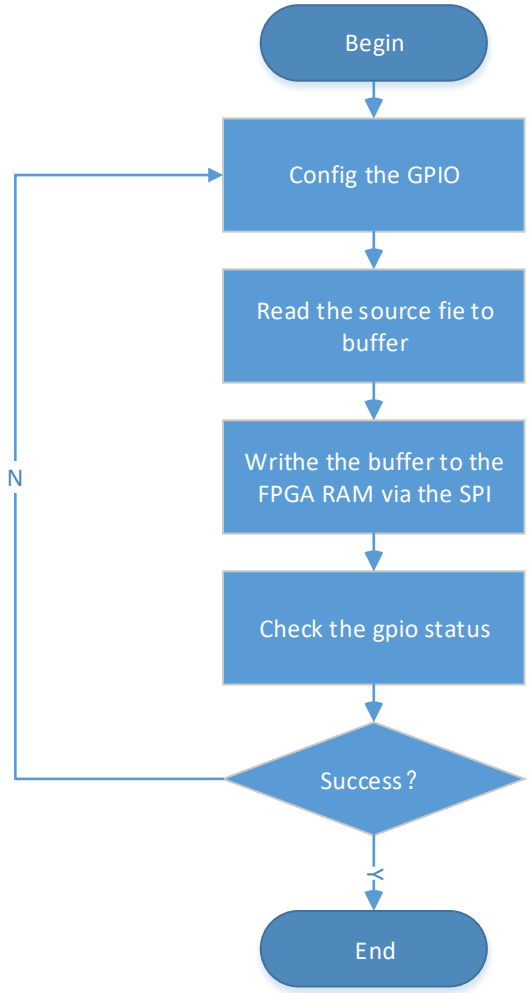

Figure 4-1 FPGA firmware update process
FPGA 固件升级

4.4 CM network storm protection CM 防网络风暴

4.4.1 gfar_clean_rx_ring

SWDD-CM-DD_NSafR_SecR_A_004

4.4.1.1 Function Description 功能描述

This function is used to determine the flow of data packets and network card shut down.

本函数用于进行数据包的流量判断以及网卡关闭。

4.4.1.2 Argument Description 参数说明

- Function Definition 函数定义

int gfar_clean_rx_ring(struct gfar_priv_rx_q *rx_queue, int rx_work_limit)

- Input argument 输入参数

rx_queue: Net card's receive queue. 网卡的接收队列。

rx_work_limit: The maximum number of packets a receive operation can receive. 一次接收操作能接收的最多包数。

- Output argument 输出参数

Return the number of received packets, 返回接收的数据包个数。

4.4.1.3 处理流程

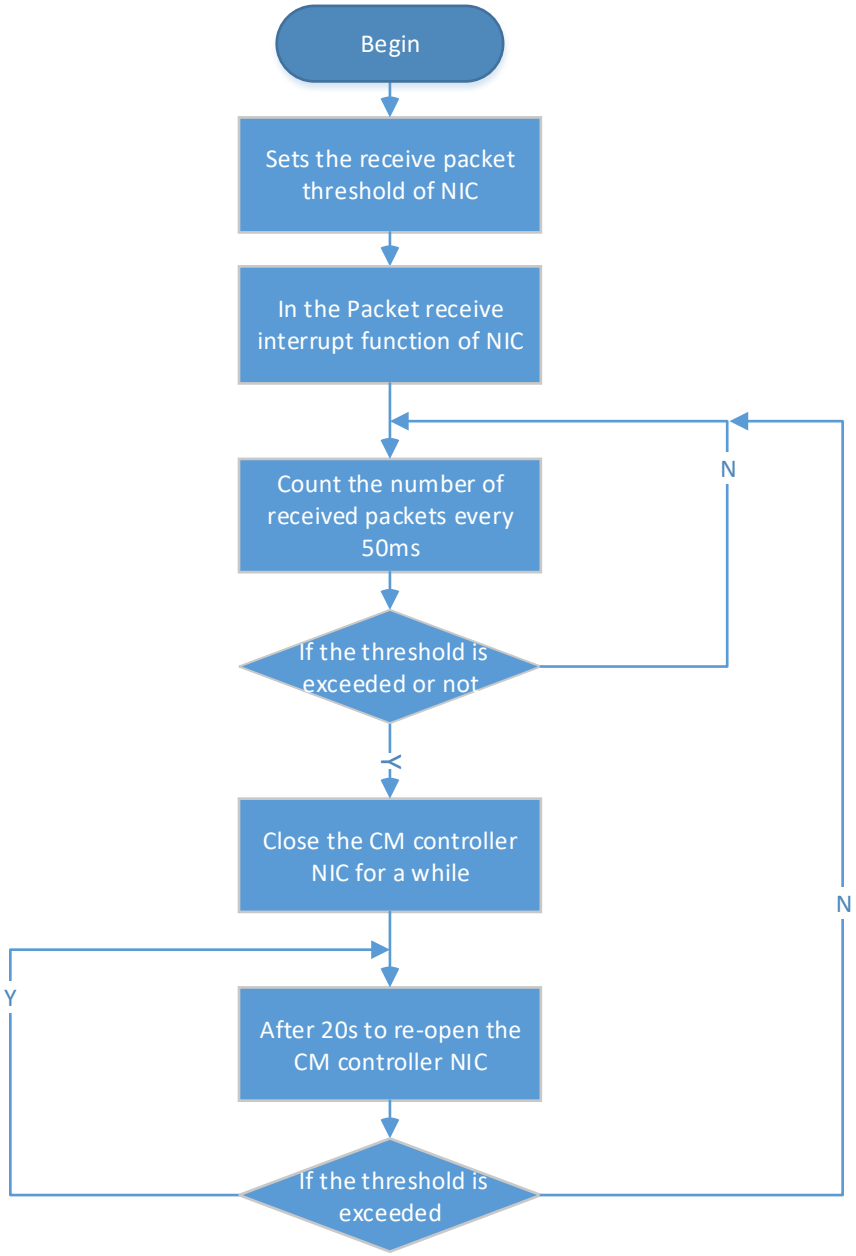

Figure 4-2 network storm protection
防网络风暴流程图

4.5 SRAM driver SRAM 驱动

SWDD-CM-DD_NSafR_NSecR_A_005

4.5.1 sram_read

4.5.1.1 Function Description 功能描述

This function is used to read SRAM.

本函数用来读 SRAM。

4.5.1.2 Argument Description 参数说明

- Function Definition 函数定义

static ssize_t sram_read(struct file *file, char __user *buf,size_t count, loff_t *ppos)

- Input argument 输入参数

file: File pointer, 文件指针。

buf: User data buffer, 用户数据缓存。

count: The number of bytes to read, 要读的字节数。

ppos: SRAM's operation offset, SRAM 的操作偏移量。

- Output argument 输出参数

Return 0 and negative for failed operation, 返回负数表示操作失败。

Returns the number of bytes, 返回读取的字节数。

4.5.1.3 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。

4.5.2 sram_write

4.5.2.1 Function Description 功能描述

This function is used to write SRAM.

本函数用于写 SRAM。

4.5.2.2 Argument Description 参数说明

- Function Definition 函数定义

static ssize_t sram_write(struct file *file, const char __user *buf, size_t count, loff_t *ppos)

- Input argument 输入参数

file: File pointer, 文件指针。

buf: User data buffer, 用户数据缓存。

count: The number of bytes to write, 要写的字节数。

ppos: SRAM's operation offset, SRAM 的操作偏移量。

- Output argument 输出参数

Return 0 and negative for failed operation, 返回负数表示操作失败。

Returns the number of bytes, 返回写入的字节数。

4.5.2.3 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。

——以下无正文