Document Title: PM_FW Schedule Module design description of Safety Control System

Document Number: 17-Q04-000618

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 PM_FW 调度模块设计说明书

文件编号：17-Q04-000618

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 task schedule 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 schedule 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 任务调度模块的设计方案。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 任务调度模块设计的输出，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
14.	TCB	Task control block	任务控制块

2 Module overview 模块概述

The location of the schedule module (marked red) in the software hierarchy is shown below.

调度模块（标红）在软件层次中的位置如下图所示。

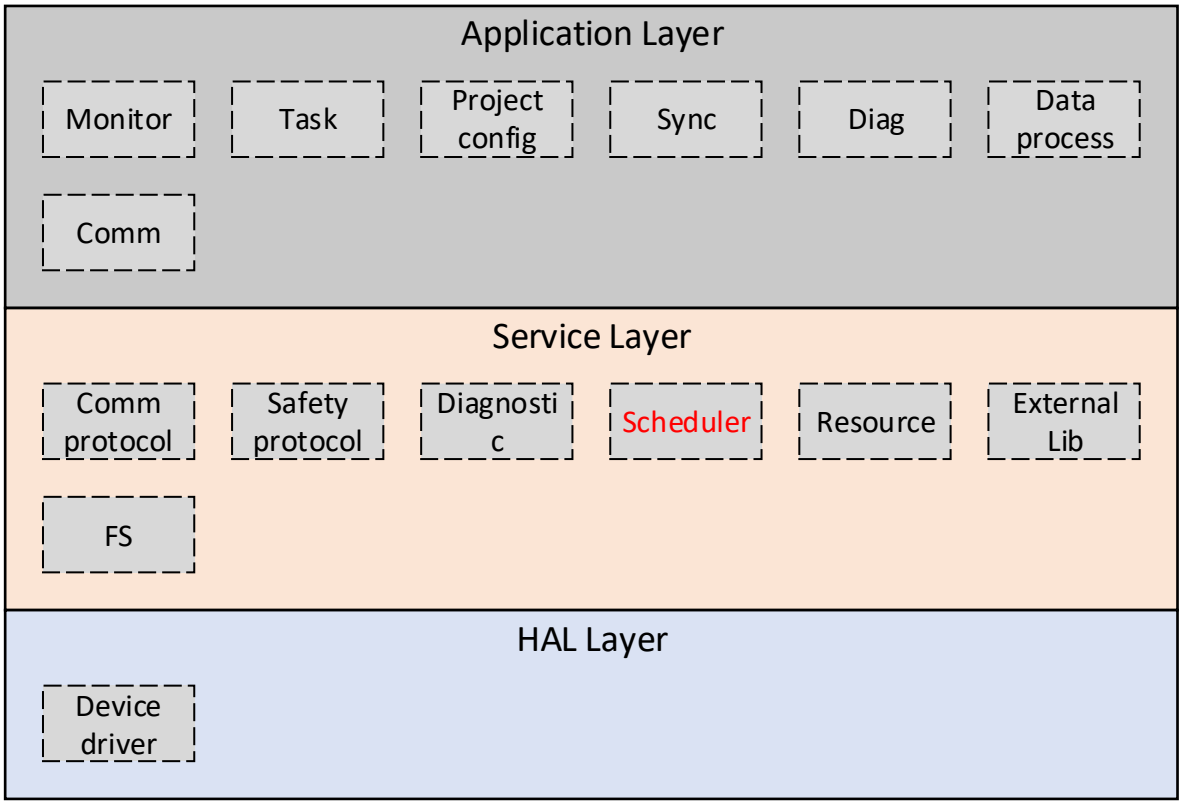

Figure 2-1 the location of the module

图 2-1 模块位置

Schedule module is used to assign the running time of each task, and responsible for task switching.

任务调度模块主要是用于分配各任务的运行时间，以及任务切换。

3 Module design 模块设计

3.1 Function description 功能描述

This document describes three sub-functions, task management, task scheduling, and task switching.

本文档主要描述三个子功能，任务管理，任务调度，任务切换。

3.2 Design concept 设计思路

3.2.1 TASK Management TASK 管理

The design process of a real-time application generally involves splitting the work to be completed into tasks, and each task is responsible for a portion of the problem. A task (also called a thread) is a simple program that thinks it has the Central Processing Unit (CPU) all to itself. On a single CPU, only one task can execute at any given time.

实时应用设计过程中通常将工作分成几个任务，每个任务负责问题的一部分，任务（也称为线程）是一个简单的程序，运行在在单个 CPU 上，在任何给定时间内只能执行一个任务。

For each task, a structure is defined to describe the task. The CPU execution context of the task describes the state of the processor running and the resource occupation of the task. Task management refers to the creation of a new task, and the steps of creating a new task is as follows:

1. Set the task priority.
2. Set the task stack.
3. Initialize task control block.
4. Load the context of task into CPU.

As shown in Figure 3-2-1, after creation of step 1, 2, 3, and 4, each task has a task entry function. When the task runs, the register state of the CPU is called task context.

对于每个任务，定义一个结构来对任务进行描述，任务的 CPU 运行上下文描述了任务在处理器执行过程中的，处理器运行状态和任务占用资源的情况，任务管理指的是创建一个新的任务，创建一个新任务的过程如下：

1. 设置任务优先级。
2. 设置任务堆栈。
3. 初始化任务控制块。
4. 装载任务运行。

如图 3-1 所示，1,2,3,4 创建完成之后，每个任务有一个任务入口函数，任务运行时 CPU 的寄存器状态就称为任务上下文。

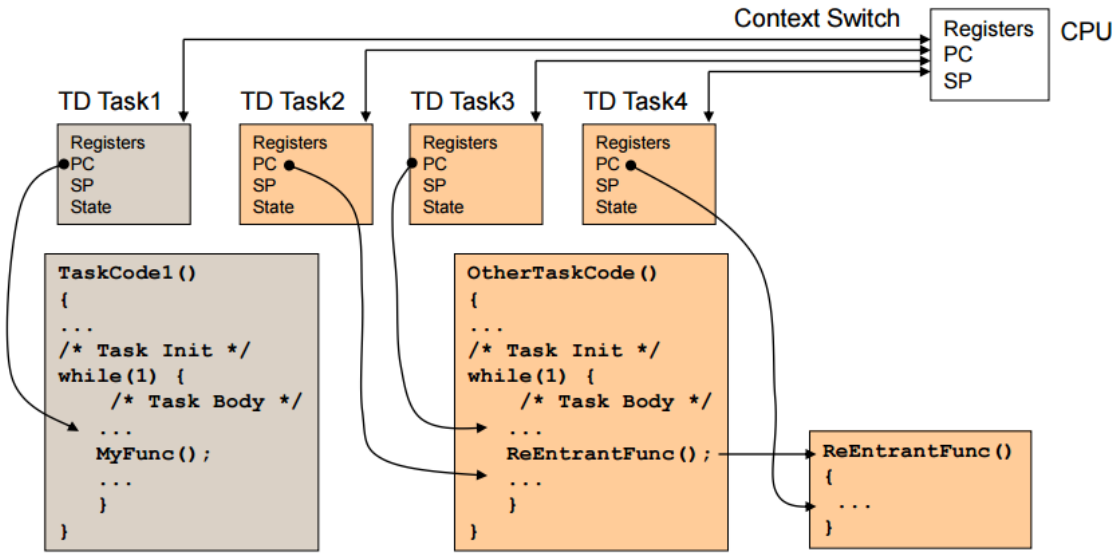

Figure 3-1 Creation of tasks

图 3-1 任务创建

3.2.2 TASK Scheduling TASK调度

Task scheduling determines the order of running tasks, and each task is assigned a priority that supports multiple tasks with the same priority. When an event occurs, the task is stopped if the event is ready for a high priority task, higher priority tasks get control of the CPU. When multiple tasks have the same priority, the system allows each task to run for the specified time slice. Of course, when the task does not run out of time slots allocated for it, it can give up CPU voluntarily; when the allocated time slice runs out, the system stops running the current task and switches to the next equal priority task to continue running.

任务调度决定了任务的运行顺序，每个任务都被分配了一个优先级，支持多任务有相同的优先级，当事件发生时候，如果事件让高优先级的任务就绪，当前任务就停止运行，更高优先级的任务获得 CPU 的控制权，当多个任务有相同的优先级时候，系统允许每个任务运行规定的时间片，当然任务没有用完分配给它的时间片时候，它可以自愿放弃 CPU，当分配的时间片用完时候，系统停止运行当前任务，切换下一个同等优先级的任务继续运行。

When two or more tasks have the same priority, system allows one task to run for a predetermined amount of time (called a Time Quanta) before selecting another task. This process is called Round-Robin scheduling or Time Slicing. If a task does not need to use its full time quanta it can voluntarily give up the CPU so that the next task can execute. System allows the user to enable or disable round robin scheduling at run time.

当两个或多个任务具有相同的优先级时，系统允许一个任务在选择另一个任务之前运行

预定的时间量(称为 time quanta)。 此过程称为循环调度或时间轮转调度。如果任务不需要使用其全部时间片，它可以自愿放弃 CPU，以便下一个任务可以执行。允许用户在运行时启用或禁用循环调度。

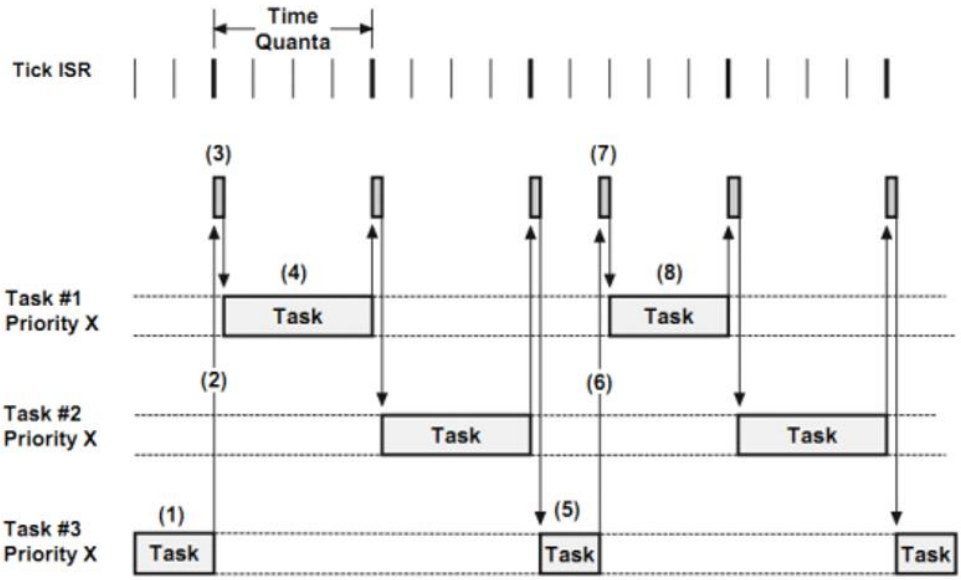

Figure 3-2 the Round Robin scheduling

图 3-2 任务轮询

The figure below shows a timing diagram with tasks running at the same priority. There are three tasks that are ready-to-run at priority “X”. For sake of illustration, the time quanta occurs every 4th clock tick. This is shown as a darker tick mark.

图 3-2 是相同优先级下任务运行的时序图，如图，有三个优先级都为 X 的任务，时间片长度都为 4

(1) Task 3 is running, during this time, the clock interrupt occurs, but the task 3 has not finished using the time slice,

(2) Task 3 automatically gives up the time slice.

(3) System Recovery Task 1, because it is the next task in Task 3 of the queue.

(4) Task 1 is executed until the time slice assigned to it expires.

(5) Task 3 is being executed, during that time the clock interrupt occurs, but task 3 has not expired.

(6) Task 3 automatically gives up the time slice.

(7) System reset the time slice of task 1 to 4.

(8) Task 1 Execution.

(1) 任务 3 正在被运行，在这段时间内，时钟中断发生，但是任务 3 还没有使用完时间片

(2) 任务 3 主动放弃时间片，

(3) 系统恢复任务 1，因为它是队列中任务 3 的下一个任务。

(4) 任务 1 被执行直到分配给它的时间片到期。

(5) 任务 3 正在被执行，在这段时间内，时钟中断会发生，但任务 3 还没有到期。

(6) 任务 3 主动放弃剩下的时间片。

(7) 系统重新设置任务 1 的时间片为 4。

(8) 任务 1 执行。

3.2.3 TASK Switching TASK切换

When the system turns to perform another task, it saves the current task's registers to the task stack and restores the context from the stack of other task to the CPU. This process is called task switching.

当系统转向去执行另一个任务的时候，它保存了当前的任务的寄存器到任务堆栈，并从其它任务的堆栈中恢复上下文到 CPU，这个过程叫任务切换。

An interrupt is a hardware mechanism used to inform the CPU that an asynchronous event occurred. When an interrupt is recognized, the CPU saves part (or all) of its context (i.e., registers) and jumps to a special subroutine called an Interrupt Service Routine (ISR). The ISR processes the event, and after completion of the ISR, the program either returns to the interrupted task, or the highest priority task, if the ISR made a higher priority task ready-to-run.

中断是硬件机制，用于通知 CPU 有异步事件发生，当中断被响应时，CPU 保存部分或者全部寄存器到任务的堆栈里，并跳转到中断服务程序 ISR 里，ISR 处理完成之后，程序会返回到中断前的任务或者更高优先级的任务。

If the high-priority task B is ready in the interrupt handler, the interrupt function does not return to the pre-interrupt task A, but directly executes the high-priority task B, at this time, due to interrupt generation, the task A State is stored in the task A's stack, so when the interrupt returns the state of A is not saved again but directly load the task B context to the hardware CPU.

如果高优先级任务 B 就绪了在中断处理函数里，中断函数返回时候不会回到中断前的任务 A，而是直接执行高优先级任务 B，此时，由于中断产生时，已经将任务 A 的状态保存在任务 A 的堆栈里，所以中断返回时无需再保存 A 的状态，直接载入任务 B 的上下文到硬件 CPU 中。

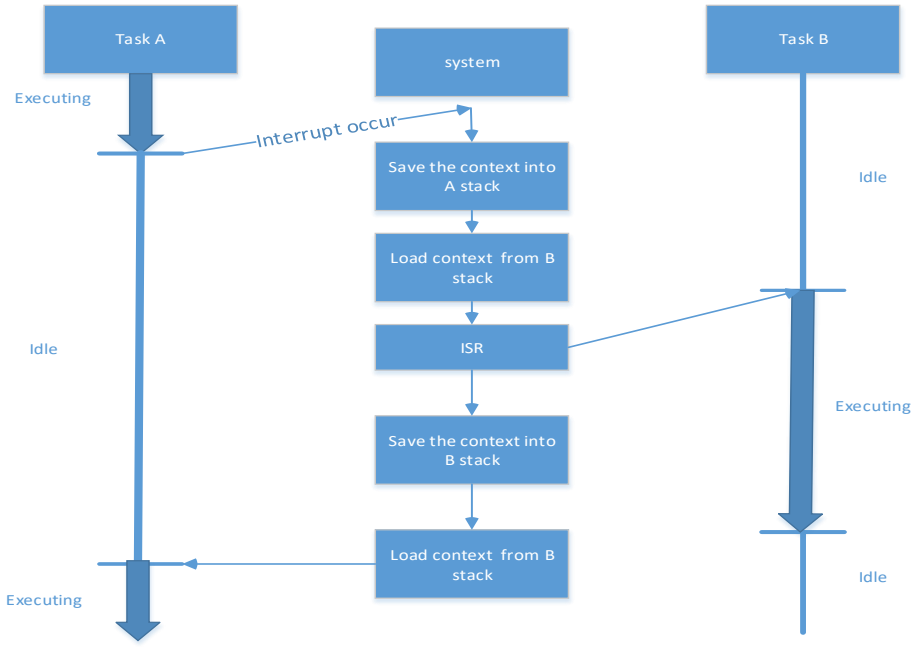

Figure 3-3 Task switching timing diagram

图 3-3 任务切换

3.3 Interface function 接口函数

3.3.1 TASK Management TASK管理

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. int os_create_task(char *pname, void *ptask, void *parg, int prio,unsigned int tq, unsigned int *err)

Input argument 输入参数	Output argument 输出参数	Description 描述
char *pname, void *ptask, void *parg, int prio,unsigned int tq, unsigned int *err pname:任务名 ptask :任务函数 parg:函数参数 prio:任务优先级 tq:时间片个数 err:返回操作错误码	Task id. 任务 id	Create a new task 创建任务

3.3.2 TASK Scheduling TASK调度

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 os_scheduled(void)

Input argument 输入参数	Output argument 输出参数	Description 描述
No. 无。	No. 无。	Context schedule. 任务调度

2. void os_sched_rr(void)

Input argument 输入参数	Output argument 输出参数	Description 描述
No. 无。	No. 无。	Round Robin schedule. 轮转调度

3.3.3 TASK Switching TASK切换

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 os_int_sw(void)

Input argument 输入参数	Output argument 输出参数	Description 描述
No. 无。	No. 无。	Context switching. 上下文切换

2. void os_time_tick(void)

Input argument 输入参数	Output argument 输出参数	Description 描述
No. 无。	No. 无。	ISR 中断服务程序

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	unsigned long	os_prio_cur	Priority of current task 当前任务的优先级 即将要被挂起的任务的优先级
2	unsigned long	os_prio_highrdy	Priority of highest priority task 指向即将要运行的任务的优先级
3	os_tcb *	os_tcb_cur	Pointer to currently running TCB 指向当前运行任务的控制块指针
4	os_tcb *	os_tcb_highrdy	Pointer to highest priority TCB R-to-R 指向切换后运行的任务的任务控制块
5	os_tcb *	os_tcb_list	Pointer to doubly linked list of TCBs 任务控制块链表，指向最后最新建立的任务
6	os_tcb *	os_tcb_prio_tbl[32]	Table of pointers to created TCBs 任务控制块优先级链表指针指向各个优先级的任务控制块
7	os_tcb	os_tcb_tbl[32]	Table of TCBs 任务控制块数组

3.5 Data structure 数据结构

```
struct os_tcb {
    INT32U      *StkPtr;        /* Pointer to current top of stack      */
    void        *ExtPtr;        /* Pointer to user definable data for TCB extension */
    OSTCB       *NextPtr;        /* Pointer to next TCB in the TCB list    */
```

```
OSTCB       *PrevPtr;          /* Pointer to previous TCB in the TCB list */
OSTCB       *TickNextPtr;
OSTCB       *TickPrevPtr;
OSTICK_LIST     *TickListPtr;           /* Pointer to tick list if task is in a tick list   */
INT32U          *StkBasePtr;            /* Pointer to base address of stack */
OSPEND_DATA    *PendDataTblPtr;        /* Pointer to list containing objects pended on */
INT32U          PendOn;                 /* Indicates what task is pending on */
INT32U          PendStatus;             /* Pend status    */
INT32U          Prio;                   /* Task priority (0 == highest)*/
INT32U          StkSize;            /* Size of task stack (in number of stack elements) */
INT32U          Opt;                    /* Task options as passed by TaskCreate() */
OS_QTY          PendDataTblEntries;    /* Size of array of objects to pend on */
INT32U          TS;                     /* Timestamp    */
INT32U          SemCtr;                 /* Task specific semaphore counter*/
INT32U          TickRemain;             /* Number of ticks remaining */
INT32U          TimeQuanta;
INT32U          TimeQuantaCtr;
INT32U          Tconsuming;
char             *t_name;
};

struct    os_rdy_list {
      OSTCB       *HeadPtr;          /* Pointer to task that will run at selected priority */
      OSTCB       *TailPtr;           /* Pointer to last task at selected priority */
      OSQTY       NbrEntries;         /* Number of entries at selected priority */
};
```

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-PM-TS_SafR_NSecR_A_001	TASK Management TASK 管理
SWDD-PM-TS_SafR_NSecR_A_002	TASK scheduling TASK 调度
SWDD-PM-TS_SafR_NSecR_A_003	TASK switching TASK 切换

4 Design of sub-function 子功能设计

4.1 Task management Task 管理

SWDD-PM-TS_SafR_NSecR_A_001

4.1.1 os_create_task

4.1.1.1 Function Description 功能描述

This function is used to create task.

本函数用于任务的创建。

4.1.1.2 Argument Description 参数说明

- Function Definition 函数定义

int os_create_task(char *pname, void *ptask, void *parg, int prio,unsigned int tq, unsigned int *err)

- Input argument 输入参数

pname：task name, 任务名，ptask：task entry function, 任务函数，parg：function parameter 函数参数，prio：task priority 任务优先级，tq：time slice 时间片个数，err：error code 返回操作错误码

- Output argument 输出参数

Task id：task identity 任务 id

4.1.1.3 Processing flow 处理流程

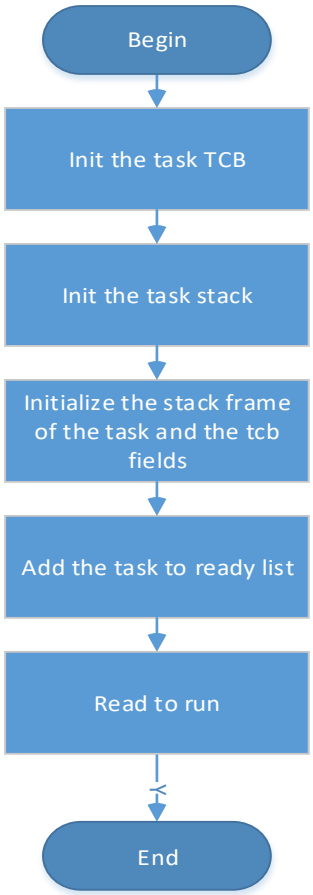

Figure 4-1-1-3 Task create process
任务创建流程

4.2 Task Schedule 任务调度

4.2.1 OSSchedule

SWDD-PM-TS_SafR_NSecR_A_002

4.2.1.1 Function Description 功能描述

This function is called by system services to determine whether a new, higher priority task has been made ready to run.

此函数由系统服务调用以确定新的高优先级任务是否已准备好运行。

4.2.1.2 Argument Description 参数说明

- Function Definition 函数定义

void os_scheduled(void)

- Input argument 输入参数

No.

无。

➢ Output argument 输出参数

No.

无。

4.2.1.3 处理流程

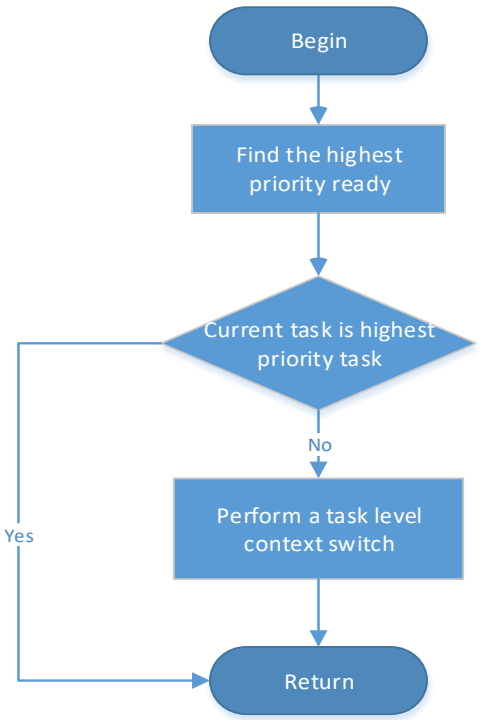

Figure 4-2-1-3 Task Schedule process
任务调度流程

4.2.2 os_sched_rr

SWDD-PM-TS_SafR_NSecR_A_002

4.2.2.1 Function Description 功能描述

This function is called on every tick ISR to determine if a new task of the same priority needs to execute.

在每个 Tick 的时钟中断处理函数上调用此函数，以确定是否需要执行相同优先级的新任务。

4.2.2.2 Argument Description 参数说明

➢ Function Definition 函数定义

void os_sched_rr(void)

➢ Input argument 输入参数

No.

无。

➢ Output argument 输出参数

No.

无。

4.2.2.3 Processing flow 处理流程

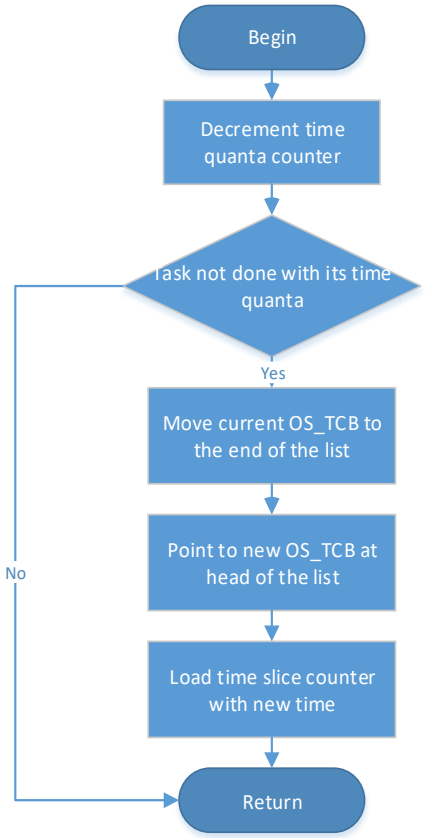

Figure 4-2-1-3 Task round robin schedule process
任务轮转调度过程

4.3 Task Switching Task 切换

SWDD-PM-TS_SafR_NSecR_A_003

4.3.1 os_int_sw

4.3.1.1 Function Description 功能描述

This function is used to perform task switching in the interrupt handler.

本函数用于进行任务切换在中断处理函数里。

4.3.1.2 Argument Description 参数说明

➢ Function Definition 函数定义

void os_int_sw(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

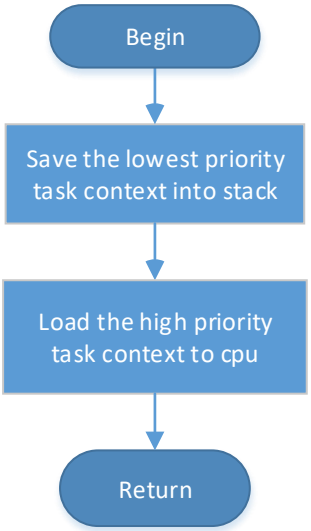

Figure 4-3-1-3 Task context switch process
任务上下文切换过程

4.3.2 os_time_tick

4.3.2.1 Function Description 功能描述

This function is used to signal to system the occurrence of a 'system tick' (also known as a 'clock tick').

本函数用于向系统发送“系统 tick”(也称为”时钟 tick”)，此函数在系统中时钟断函数里调用。

4.3.2.2 Argument Description 参数说明

➢ Function Definition 函数定义

void os_time_tick(void)

➢ Input argument 输入参数

No.

无。

- Output argument 输出参数

No.

无。

4.3.2.3 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。

——以下无正文