



|  |  |
| --- | --- |
| IPO表 | |
| 系统：CNS多任务系统 | 作者：Ljf |
| 模块：数据传输中断（CMN\_Int） | 日期：2018/01/10 |
| 编号： |  |
| 被调用： | 调用：  ND\_OSIntEntry(),ND\_OSTaskQPost(\*),  ND\_OSIntExit() |
| 输入：  PC端传输过来的数据 | 输出：   1. 把数据放入数据处理任务的消息队列 2. 唤醒错误处理任务 |
| 处理：  ND\_OSIntEntry();  把数据放入CMN\_buf数据缓存区;  If(数据传输结束)  ND\_OSTaskQPost(“数据处理任务消息队列”,Posteer)  Switch(Posterr)  {  Case: OVER break;  Case: NONE break;  Case:OTHER ND\_OSTaskQPost(“错误处理任务”);break;  }  ND\_OSIntExit(); | |
| 局部数据元素：  CMN\_buf数据缓存区  Posterr | 注释： |
| 所用堆栈：  与其他中断共享一个中断堆栈  中断内几个个函数的返回地址，函数内的还会有一些变量，大概占用100byte不到的堆栈，但是考虑到中断嵌套的情况，分配中断堆栈1024byte。 | |

|  |  |
| --- | --- |
| IPO表 | |
| 系统：CNS多任务系统 | 作者：Ljf |
| 模块：数据处理任务（DataCheckTask） | 日期：2018/01/10 |
| 编号： |  |
| 被调用： | 调用：  ND\_OSTaskQPost(\*),ND\_DataErrorCheck(\*),  ND\_OSTaskSemPost(\*),ND\_OSTaskQPend(\*) |
| 输入：  从数据处理任务消息队列中获取PC数据 | 输出：   1. 通知PC机错误信息 2. 唤醒电量和人脑检测任务 3. 唤醒刺激任务 4. 唤醒Write Flash 任务 |
| 处理：  ND\_OSTaskQPend(“数据处理任务消息队列”,Pender);  Switch(Penderr)  {  Case: EXIST  ND\_DataErrorCheck(\*,Dataerr);  Switch(Dataerr)  {  Case:ERROR 通知PC数据错误，请重发; break;  Case:IMPCHK ND\_OSTaskSemPost(“电量和人脑阻抗检测”);break;  Case:STM ND\_OSTaskQPost(“刺激任务”);break;  Case:WRF ND\_OSTaskQPost(“Write Flash 任务”);break;  }  break;  Case: NONE 任务挂起；break;  Case:OTHER ND\_OSQPost(“错误处理任务”);break;  } | |
| 局部数据元素：  Penderr  DataCheck\_buf 处理数据的缓存区 | 注释： |
| 所用堆栈：  每个任务都有单独的堆栈  堆栈保存寄存器所用空间140byte+局部变量+函数返回地址<300byte  所以分配512byte | |

|  |  |
| --- | --- |
| IPO表 | |
| 系统：CNS多任务系统 | 作者：Ljf |
| 模块：Write Flash任务（WriteFlahsTask） | 日期：2018/01/10 |
| 编号： |  |
| 被调用： | 调用：  ND\_OSTaskQPost(\*),ND\_WriteFlash(\*),  ND\_OSTaskQPend(\*) |
| 输入：  从Write Flash任务消息队列中获取PC数据 | 输出：   1. 把相应数据写入Flash，通知PC写入情况 2. 任务挂起 3. 错误处理任务 |
| 处理：  ND\_OSTaskQPend(“Write Flash 任务”,Penderr);  Swich(Penderr)  {  Case: EXIST  Writeerr=ND\_WriteFlash(\*);  Swich(Writeerr)  {  Case:ERROR 通知PC数据写入错误 break;  Case:SUCCESS 通知PC数据写入成功 break；  }  break;  Case: NONE 任务挂起；break;  Case:OTHER ND\_OSQPost(“错误处理任务”);break;  } | |
| 局部数据元素：  Penderr  WriteData\_buf 需要写入数据的缓存区 | 注释： |
| 所用堆栈：  堆栈寄存器所用空间140byte+局部变量+函数返回地址<300byte  所用分配512byte | |

|  |  |
| --- | --- |
| IPO表 | |
| 系统：CNS多任务系统 | 作者：Ljf |
| 模块：刺激任务（StimulateTask） | 日期：2018/01/10 |
| 编号： |  |
| 被调用： | 调用：  ND\_OSTaskQPost(\*),ND\_Stimulate(\*),  ND\_OSTaskQPend(\*) |
| 输入：  从刺激任务消息队列中获取刺激数据 | 输出：   1. 进行刺激 2. 任务挂起 3. 唤醒错误处理任务 |
| 处理：  ND\_OSTaskQPend(“刺激任务”,Penderr);  Swich(Penderr)  {  Case: EXIST ND\_Stimulate(\*);break;  Case: NONE 任务挂起；break;  Case:OTHER ND\_OSQPost(“错误处理任务”);break;  } | |
| 局部数据元素：  Penderr  STM\_buf 刺激数据缓存区 | 注释： |
| 所用堆栈：  堆栈寄存器所用空间140byte+局部变量+函数返回地址<300byte  所用分配512byte | |

|  |  |
| --- | --- |
| IPO表 | |
| 系统：CNS多任务系统 | 作者：Ljf |
| 模块：Read Flash任务（ReadFlashTask） | 日期：2018/01/10 |
| 编号： |  |
| 被调用： | 调用：  ND\_OSTaskQPost(\*),  ND\_ReadFlash(\*Read\_buf),  ND\_OSTaskQPend(\*) |
| 输入：  从Read Flash任务消息队列中获取读取指令 | 输出：   1. 把数据传输给上位机 2. 任务挂起 3. 唤醒错误处理任务 |
| 处理：  ND\_OSTaskQPend(“Read Flash任务”,Penderr);  Swich(Penderr)  {  Case: EXIST ND\_ReadFlash(\*Read\_buf) 传输数据给上位机;break;  Case: NONE 任务挂起；break;  Case:OTHER ND\_OSQPost(“错误处理任务”);break;  } | |
| 局部数据元素：  Read\_buf 数据缓存区  ReadCmd\_buf 读指令缓存区  Penderr | 注释： |
| 所用堆栈：  堆栈寄存器所用空间140byte+局部变量+函数返回地址<300byte  所用分配512byte | |

|  |  |
| --- | --- |
| IPO表 | |
| 系统：CNS多任务系统 | 作者：Ljf |
| 模块：电量和人脑阻抗检测（EleAndImpTask） | 日期：2018/01/10 |
| 编号： |  |
| 被调用： | 调用：  ND\_OSTaskQPost(\*),ND\_OSTaskSemPend(\*) |
| 输入：  从电量和人脑检测的信号量中读取信号量 | 输出：   1. 把数据传输给PC 2. 任务挂起 3. 唤醒错误处理任务 |
| 处理：  ND\_OSTaskSemPend(“电量和人脑阻抗检测任务”,Penderr);  Swich(Penderr)  {  Case: EXIST 读取电量和人脑阻抗检测数据 传输数据给上位机;break;  Case: NONE 任务挂起；break;  Case:OTHER ND\_OSQPost(“错误处理任务”);break;  } | |
| 局部数据元素：  Penderr | 注释： |
| 所用堆栈：  堆栈寄存器所用空间140byte+局部变量+函数返回地址<300byte  所用分配512byte | |

|  |  |
| --- | --- |
| IPO表 | |
| 系统：CNS多任务系统 | 作者：Ljf |
| 模块：充电任务（ChargeTask） | 日期：2018/01/10 |
| 编号： |  |
| 被调用： | 调用：  ND\_OSTaskQPost(\*),ND\_OSTaskSemPend(\*),  ND\_OSSendNbrEle() |
| 输入：  从充电任务的信号量中读取信号量 | 输出：   1. 使发送电量信息使能位置1 2. 任务挂起 3. 唤醒错误处理任务 |
| 处理：  ND\_OSTaskSemPend(“充电任务”,Penderr);（此函数不减少“充电”信号量）  Swich(Penderr)  {  Case: EXIST ND\_OSSendNbrEle();  If(是否断开充电) {“充电”信号量赋值为0;}break;  Case: NONE 任务挂起；break;  Case:OTHER ND\_OSQPost(“错误处理任务”);break;  } | |
| 局部数据元素：  Penderr | 注释： |
| 所用堆栈：  堆栈寄存器所用空间140byte+局部变量+函数返回地址<300byte  所用分配512byte | |

|  |  |
| --- | --- |
| IPO表 | |
| 系统：CNS多任务系统 | 作者：Ljf |
| 模块：电量不足通知（NoticeEleTask） | 日期：2018/01/10 |
| 编号： |  |
| 被调用： | 调用：  ND\_OSTaskQPost(\*),ND\_OSTaskSemPend(\*) |
| 输入：  从通知任务的信号量中读取信号量 | 输出：   1. 发送电量不足和电量信息给上位机 2. 唤醒充电任务 3. 停止采样任务 4. 唤醒错误处理任务 |
| 处理：  ND\_OSTaskSemPend(“电量不足任务”,Penderr) （此函数不减少“电量不足”信号量）  Swich(Penderr)  {  Case: EXIST 发送电量不足和电量信息给上位机;  If(连上充电器) ND\_OSSemPost(“充电任务”); “电量不足”信号量赋值为0;  If(电量低于临界值) ND\_OSSemPost(“停止采样任务”);“电量不足”信号量赋值为0;break;  Case: NONE 任务挂起；break;  Case:OTHER ND\_OSQPost(“错误处理任务”);break;  } | |
| 局部数据元素：  Penderr | 注释： |
| 所用堆栈：  堆栈寄存器所用空间140byte+局部变量+函数返回地址<300byte  所用分配512byte | |

|  |  |
| --- | --- |
| IPO表 | |
| 系统：CNS多任务系统 | 作者：Ljf |
| 模块：ADC采样和异常检测（SampleAndErrChkTask） | 日期：2018/01/10 |
| 编号： |  |
| 被调用： | 调用：  ND\_OSTaskQPost(\*),ND\_OSTaskSemPend(\*),  ND\_GetSample(\*Data\_buf),  ND\_ErrorCheck(\*Data\_buf) |
| 输入：  从停止采样任务的信号量中读取信号量 | 输出：   1. 挂起任务 2. 唤醒刺激任务 3. 继续采样 4. 唤醒错误处理任务 |
| 处理：  ND\_OSTaskSemPend(“停止采样任务”,Pender);  Swich(Penderr)  {  Case: EXIST 任务挂起；break;  Case: NONE ND\_GetSample(\*Data\_buf);  Eegerr=ND\_ErrorCheck(\*Data\_buf)  Swich(Eegeer)  {  Case:EXIST\_EER ND\_OSTaskQPost(“刺激任务”); break;  Case:NON\_EXIST\_EER break;  }  Case:OTHER ND\_OSQPost(“错误处理任务”);break;  } | |
| 局部数据元素：  Data\_buf 采样数据缓存区  Penderr | 注释： |
| 所用堆栈：  堆栈寄存器所用空间140byte+局部变量+函数返回地址<300byte  所用分配512byte | |

|  |  |
| --- | --- |
| IPO表 | |
| 系统：CNS多任务系统 | 作者：Ljf |
| 模块：休眠任务(SleepModeTask) | 日期：2018/01/10 |
| 编号： |  |
| 被调用： | 调用：  ND\_OSTaskCreate(\*),ND\_OSEnterSleepMode() |
| 输入： | 输出：   1. 创建刺激任务 2. 创建Write Flash任务 3. 创建充电任务 4. 创建Read Flash任务 5. 创建通知电量不足任务 6. 创建数据检测任务 7. 创建错误处理任务 8. 创建ADC采样和异常检测任务 9. 进入休眠模式 |
| 处理：  Createerr=ND\_OSTaskCreate(“刺激任务”);  Createerr=ND\_OSTaskCreate(“Write Flash任务”);  Createerr=ND\_OSTaskCreate(“充电任务”);  Createerr=ND\_OSTaskCreate(“电量和人脑阻抗任务”);  Createerr=ND\_OSTaskCreate(“Read Flash任务”);  Createerr=ND\_OSTaskCreate(“通知电量不足任务”);  Createerr=ND\_OSTaskCreate(“数据检测任务”);  Createerr=ND\_OSTaskCreate(“错误处理任务”);  Createerr=ND\_OSTaskCreate(“ADC采样和异常检测”);  ND\_OSEnterSleepMode(); | |
| 局部数据元素：  Createerr | 注释： |
| 所用堆栈：  堆栈寄存器所用空间140byte+局部变量+函数返回地址<300byte  所用分配512byte | |

|  |  |
| --- | --- |
| IPO表 | |
| 系统：CNS多任务系统 | 作者：Ljf |
| 模块：错误处理任务（ManageErrorTask） | 日期：2018/01/10 |
| 编号： |  |
| 被调用： | 调用：  ND\_OSTaskQPend(\*), |
| 输入：  从错误处理任务的消息队列中获取错误信息 | 输出：  1.通知PC或者  通知头戴设备或者  通知手持设备  2.任务挂起 |
| 处理：  ND\_OSTaskQPend(“错误处理任务”,Penderr);  Swich(Penderr)  {  Case: EXIST 通知PC或者通知头戴设备或者通知手持设备;break  Case: NONE 任务挂起；break;  Case:OTHER 通知PC或者通知头戴设备或者通知手持设备;break  } | |
| 局部数据元素：  Penderr  DataErr\_buf 错误数据缓存区 | 注释： |
| 所用堆栈：  堆栈寄存器所用空间140byte+局部变量+函数返回地址<300byte  所用分配512byte | |

|  |  |
| --- | --- |
| IPO表 | |
| 系统：CNS多任务系统 | 作者：Ljf |
| 模块：温度和电量检测中断(TemAndEle\_Int) | 日期：2018/01/10 |
| 编号： |  |
| 被调用： | 调用：  ND\_OSTemCheck(\*NbrTem),  ND\_OSEleCheck(\*NbrEle),  ND\_GetCurPrio(\*CurPrio),  ND\_OSTaskSemPost(\*),  ND\_OSOff() |
| 输入： | 输出： |
| 处理：  ND\_OSTemCheck(\*NbrTem);  If(NbrTem大于温度临界值) ND\_OSOff();  ND\_OSEleCheck(\*CurNbrEle);  ND\_GetCurPrio(\*CurPrio);  If(CurPrio==1) return;  Else if(NbrEle小于电量充电临界值) ND\_OSTaskSemPost(“通知电量 不足任务”); | |
| 局部数据元素：  NbrTem 当前温度值  CurPrio 当前运行任务的优先级 | 注释： |
| 所用堆栈：  共用中断堆栈1024byte | |

|  |  |
| --- | --- |
| IPO表 | |
| 系统：CNS多任务系统 | 作者：Ljf |
| 模块：安全中断(Safe\_Int) | 日期：2018/01/10 |
| 编号： |  |
| 被调用： | 调用：  ND\_OSSafeCheck();  ND\_OSOff(); |
| 输入： | 输出： |
| 处理：  ND\_OSSafeCheck();  If(是否为强制关闭信号) ND\_OSOff(); | |
| 局部数据元素： | 注释： |
| 所用堆栈：  共用中断堆栈1024byte | |

|  |  |
| --- | --- |
| IPO表 | |
| 系统：CNS多任务系统 | 作者：Ljf |
| 模块：时钟周期中断(Tick\_Int) | 日期：2018/01/10 |
| 编号： |  |
| 被调用： | 调用：  ND\_OSTimeQuantaCtrCheck(\*) |
| 输入： | 输出： |
| 处理：  ND\_OSTimeQuantaCtrCheck(\*); | |
| 局部数据元素： | 注释： |
| 所用堆栈：  共用中断堆栈1024byte | |

|  |  |
| --- | --- |
| 全局变量： | |
| CurNbrEle | 指向系统当前电量值 |
| \*ND\_TCBCurPtr | 指向当前运行的任务TCB |
| \*ND\_TCBHighRdyPtr | 指向下一个优先级更高的任务TCB |
| NDPrioTbl[ND\_CFG\_PRIO\_MAX] | 记录哪个优先级下有任务就绪 |
| NDRdyList[ND\_CFG\_PRIO\_MAX] | 返回一个ND\_OS\_RDY\_LIST数据结构 |
| ND\_OSIntNestingCtr | 中断嵌套计数器 |

|  |  |
| --- | --- |
| 宏定义 | |
| ND\_CFG\_MSG\_POOL\_SIZE | 消息队列缓冲池中ND\_OS\_MSG总数 |
| ND\_CFG\_PRIO\_MAX | 最小优先级 |

|  |  |
| --- | --- |
| 任务优先级： | |
| 优先级0 | 通知电量不足任务 |
| 优先级1 | 错误处理任务 |
| 优先级2 | 刺激任务, |
| 优先级3 | Write Flash任务，电量和人脑阻抗检测任务，Read Flash任务，数据检测任务,充电任务,ADC采样和异常检测任务 |
| 优先级4 | 休眠任务（起始任务） |

|  |  |
| --- | --- |
| 中断优先级： | |
| 优先级0 | 安全中断 |
| 优先级1 | 数据传输中断，温度和电量检测中断 |
| 优先级2 | 时钟周期中断 |

# 系统所用内存大小（4字节对齐）

1. 中断堆栈(1024byte)
2. 数据处理任务堆栈( 512 byte ) + ND\_OS\_Q表( 16 byte) + ND\_OS\_MSG\_Q( 12 byte)+TCB( 40 byte)
3. Write Flash任务堆栈( 512 byte ) + ND\_OS\_Q表( 16 byte) + ND\_OS\_MSG\_Q( 12 byte)+TCB( 40 byte)
4. 刺激任务堆栈( 512 byte ) + ND\_OS\_Q表( 16 byte) + ND\_OS\_MSG\_Q( 12byte)+TCB( 40 byte)
5. Read Flash任务堆栈( 512 byte ) + ND\_OS\_Q表( 16 byte) + ND\_OS\_MSG\_Q( 12 byte)+TCB( 40 byte)
6. 电量和人脑阻抗检测任务堆栈( 512 byte ) + ND\_OS\_SEM表( 16 byte) +TCB( 40 byte)
7. 充电任务堆栈( 512 byte ) + ND\_OS\_SEM表( 16 byte) +TCB( 40 byte)
8. 电量不足任务堆栈( 512 byte ) + ND\_OS\_SEM表( 16 byte) +TCB( 40 byte)
9. ADC采样任务堆栈( 512 byte ) + ND\_OS\_SEM表( 16 byte) +TCB( 40 byte)
10. 休眠任务堆栈( 512 byte )+TCB( 40 byte)
11. 错误处理任务堆栈( 512 byte ) + ND\_OS\_Q表( 16 byte) + ND\_OS\_MSG\_Q( 12 byte)+TCB( 40 byte)
12. ND\_OS\_MSG\_POOL表 ( 8 byte) + 10张 (表数量暂定，可以根据需求变化) ND\_OS\_MSG表(120 byte)
13. 5张ND\_OS\_RDY\_LIST表 ( 12 byte)
14. ND\_OS\_MEM表 ( 16 byte) + 10个512byte存储块( 5120 byte) (存储块大小和数量可以根据需要调整)
15. ND\_OS\_MEM表 ( 16 byte) + 10个256byte存储块( 2560 byte) (存储块大小和数量可以根据需要调整)
16. ND\_OS\_MEM表 ( 16 byte) + 10个128byte存储块( 1280 byte) (存储块大小和数量可以根据需要调整)
17. 全局变量 CurNbrEle(4 byte)
18. 全局变量 \*ND\_TCBCurPtr( 4 byte)
19. 全局变量 \*ND\_TCBHighRdyPtr( 4 byte)
20. 全局变量 \*ND\_TCBHighRdyPtr( 4 byte)
21. 全局变量 NDPrioTbl[ND\_CFG\_PRIO\_MAX] (4 byte)
22. 全局变量 NDRdyList[ND\_CFG\_PRIO\_MAX](4 byte)

总共大小：15k+ 608 byte+代码段(不确定)

# 数据结构

|  |  |  |
| --- | --- | --- |
| 数据类型 | TCB | 任务块数据结构 |
| Int32 | \*StkPtr | 当前任务的栈顶 |
| Int32 | \*NextPtr | 下一个任务 |
| Int32 | \*PrevPtr | 上一个任务 |
| Int32 | \*NamePtr | 任务的名字 |
| Int32 | \*StkBasePtr | 任务栈的基地址 |
| Int32 | \*TaskEntryAddr | 任务代码入口地址 |
| Int32 | \*TaskEntryAry | 任务参数 |
| Int32 | \*PendDataTblPtr | 任务等待事件的对象 |
| Int8 | Pendon | 任务等待事件的类型 |
| Int8 | TaskState | 任务的状态 |
| Int8 | Prio | 优先级 |
| Int16 | StkSize | 任务栈大小 |
| Int8 | TimeQuanta | 任务时间片总数 |
| Int8 | TimeQuantaCtr | 任务时间片计数器 |

（以4字节对齐，则该表大小为40 byte）

|  |  |  |
| --- | --- | --- |
| 数据类型 | ND\_OS\_MSG\_POOL | MSG空闲列表数据结构 |
| Int32 | \*NextPtr | 缓冲池中指向下一个空闲的MSG |
| Int8 | NbrFree | 缓冲池中空闲的MSG数量 |
| Int8 | NbrUsed | 缓冲池中已使用的MSG数量 |

（以4字节对齐，则该表大小为8 byte）

|  |  |  |
| --- | --- | --- |
| 数据类型 | ND\_OS\_MSG | 消息数据结构 |
| Int32 | \*NextPtr | 指向下一个MSG |
| Int16 | MsgSize | 消息大小 |
| Int32 | \*MsgPtr | 指向存放消息地址 |

（以4字节对齐，则该表大小为12 byte）

|  |  |  |
| --- | --- | --- |
| 类型 | ND\_OS\_Q | 任务消息队列数据结构 |
| Int8 | Type | 消息队列类型 |
| Int32 | \*NamePtr | 消息队列名字 |
| Int32 | \*TCB | 消息队列所属任务的任务块 |
| Int32 | \*MsgQ | 消息队列 |

（以4字节对齐，则该表大小为16 byte）

|  |  |  |
| --- | --- | --- |
| 数据类型 | ND\_OS\_MSG\_Q | 消息队列数据结构 |
| Int32 | \*InPtr | 指向消息队列最后的MSG |
| Int32 | \*OutPtr | 指向下一个被取出的MSG |
| Int8 | NbrEntriesSize | 消息队列最大容量 |
| Int8 | NbrEntries | 已经存在的MSG数量 |

（以4字节对齐，则该表大小为12 byte）

|  |  |  |
| --- | --- | --- |
| 数据类型 | ND\_OS\_SEM | 任务信号量数据结构 |
| Int8 | Type | 信号量类型 |
| Int32 | \*NamePtr | 信号量名字 |
| Int32 | \*TCB | 信号量所属任务TCB |
| Int8 | Ctr | 信号量计数 |

（以4字节对齐，则该表大小为16 byte）

|  |  |  |
| --- | --- | --- |
| 数据类型 | ND\_OS\_MEM | 存储分区控制块 |
| Int32 | \*NamePtr | 存储分区的名字 |
| Int32 | \*FreeListPtr | 指向空闲存储块列表  （列表为单列表形式每个存储块都有指向下一个存储块的指针） |
| Int16 | BlkSize | 每个存储块大小 |
| Int16 | NbrMax | 存储块容量 |
| Int16 | NbrFree | 空闲的存储块数量 |

（以4字节对齐，则该表大小为16 byte）

|  |  |  |
| --- | --- | --- |
| 数据类型 | ND\_OS\_RDY\_LIST | 任务就绪表 |
| Int8 | NbrEntries | 任务就绪数量 |
| Int32 | \*HeadPtr | 指向下一个运行任务的TCB |
| Int32 | \*TailPtr | 指向最后一个任务的TCB |

（以4字节对齐，则该表大小为12 byte）

# 函数名

void ND\_OSIntEntry()

|  |  |
| --- | --- |
| 文件 | 调用位置 |
|  | 仅中断 |

ND\_OSIntEntry()是临界段代码，进入中断的时候会把cpu寄存器全部存入到对应的任务栈当中。

参数：

无。

返回值：

无

注意/警告：

使用的时候必须是在关中断的情况下。

void ND\_OSIntExit()

|  |  |
| --- | --- |
| 文件 | 调用位置 |
|  | 仅中断 |

ND\_OSIntExit()是临界段代码，结束中断前会把cpu寄存器restore。

参数：

无。

返回值：

无

注意/警告：

使用的时候必须是在关中断的情况下。

void ND\_OSTaskQPost(\*)

|  |  |
| --- | --- |
| 文件 | 调用位置 |
|  | 任务和中断 |

ND\_OSTaskQPost(\*)根据参数对相应的任务消息队列发消息。如果相应的消息队列已满则会丢弃当前的消息（也可以选择错误处理任务进行错误处理）。之后会调用任务调度器进行任务调度。

参数：

返回值：

无

注意/警告：

必须创建消息队列才能使用。

void ND\_OSTaskQPend(\*)

|  |  |
| --- | --- |
| 文件 | 调用位置 |
|  | 仅任务 |

ND\_OSTaskQPend(\*)去对应任务的消息队列中取出最早进入消息队列的消息。如果有消息就执行，没有就挂起任务，出错调用错误处理任务。

参数：

返回值：

无

注意/警告：

必须创建消息队列才能使用。

void ND\_DataErrorCheck(\*)

|  |  |
| --- | --- |
| 文件 | 调用位置 |
|  | 仅数据处理任务 |

ND\_DataErrorCheck(\*)把从数据处理任务的消息队列中取出来的数据进行处理检查。如果有错误就通知PC机重发信息，没有错误就根据信息唤醒相应的任务。

参数：

返回值：

无

注意/警告：

void ND\_OSTaskSemPost(\*)

|  |  |
| --- | --- |
| 文件 | 调用位置 |
|  | 任务和中断 |

ND\_OSTaskQPost(\*)根据参数对相应的任务信号量计数增加1。

参数：

返回值：

无

注意/警告：

必须创建任务信号量才能使用。

void ND\_OSTaskSemPend(\*)

|  |  |
| --- | --- |
| 文件 | 调用位置 |
|  | 仅任务 |

ND\_OSTaskSemPend(\*)读取相应任务的信号量计数，并减1。信号量计数如果大于0就运行任务，等于0就挂起任务。出错就运行错误处理任务。

参数：

返回值：

无

注意/警告：

必须创建任务信号量才能使用。

void ND\_WriteFlash(\*)

|  |  |
| --- | --- |
| 文件 | 调用位置 |
|  | 仅Write Flash任务 |

ND\_WriteFlash(\*)把从Write Flash任务的消息队列中取出来的数据进行写入Flash。不管写入成功与否都通知PC写入情况。

参数：

返回值：

无

注意/警告：

必须关中断

void ND\_Stimulate(\*)

|  |  |
| --- | --- |
| 文件 | 调用位置 |
|  | 仅刺激任务 |

ND\_Stimulate(\*)把从刺激任务的消息队列中取出来的数据进行刺激。

参数：

返回值：

无

注意/警告：

void ND\_ReadFlash(\*)

|  |  |
| --- | --- |
| 文件 | 调用位置 |
|  | 仅Read Flash任务 |

ND\_Stimulate(\*)把相应的数据从Flash中读取出来。

参数：

返回值：

无

注意/警告：

void ND\_GetSample(\*)

|  |  |
| --- | --- |
| 文件 | 调用位置 |
|  | 仅ADC采样和异常检测任务 |

ND\_GetSample(\*)获取采样数据，并放入缓存区中。

参数：

返回值：

无

注意/警告：

void ND\_ErrorCheck(\*)

|  |  |
| --- | --- |
| 文件 | 调用位置 |
|  | 仅ADC采样和异常检测任务 |

ND\_ErrorCheck(\*)对缓存区的数据进行异常检测。

参数：

返回值：

无

注意/警告：

void ND\_OSEnterSleepMode()

|  |  |
| --- | --- |
| 文件 | 调用位置 |
|  | 仅休眠任务 |

ND\_OSEnterSleepMode()使系统进入休眠状态。

参数：

返回值：

无

注意/警告：

void ND\_OSTaskCreate(\*),

|  |  |
| --- | --- |
| 文件 | 调用位置 |
|  | 仅休眠任务 |

ND\_OSTaskCreate(\*)创建一个新任务，并初始化任务栈。刚创建的任务处于就绪状态。

参数：

返回值：

无

注意/警告：

void ND\_OSTemCheck(\*),

|  |  |
| --- | --- |
| 文件 | 调用位置 |
|  | 仅温度ISR |

ND\_OSTemCheck(\*)获取当前系统温度。

参数：

返回值：

无

注意/警告：

void ND\_OSEleCheck(\*),

|  |  |
| --- | --- |
| 文件 | 调用位置 |
|  | 仅温度ISR |

ND\_OSEleCheck(\*)获取当前系统电量，并赋值给全局变量NbrEle。

参数：

返回值：

无

注意/警告：

void ND\_GetCurPrio(\*),

|  |  |
| --- | --- |
| 文件 | 调用位置 |
|  | ISR和任务 |

ND\_GetCurPrio(\*)获取当前运行任务的优先级。

参数：

返回值：

无

注意/警告：

void ND\_OSSendNbrEle(\*),

|  |  |
| --- | --- |
| 文件 | 调用位置 |
|  | 仅充电任务 |

ND\_OSSendNbrEle(\*)发送当前电量信息给上位机。

参数：

返回值：

无

注意/警告：

void ND\_OSOff()

|  |  |
| --- | --- |
| 文件 | 调用位置 |
|  | 仅ISR |

ND\_OSOff()关闭当前系统。

参数：

返回值：

无

注意/警告：

void ND\_OSSafeCheck(\*)

|  |  |
| --- | --- |
| 文件 | 调用位置 |
|  | 仅安全ISR |

ND\_OSSafeCheck(\*)检查触发中断的信号是否为强制关闭信号。

参数：

返回值：

无

注意/警告：

void ND\_OSTimeQuantaCtrCheck(\*)

|  |  |
| --- | --- |
| 文件 | 调用位置 |
|  | 仅时钟周期ISR |

ND\_OSTimeQuantaCtrCheck(\*)检查当前任务的时间片是否用完，如果没有用完，则时间片计数器减1，退出中断运行原先数据。如果已经用完，则切换到当前优先级的下一个任务，如果当前优先级没有下一个任务，则继续运行当前任务。

参数：

返回值：

无

注意/警告：

# 关系图









