以下部分描述 RTX51 Tiny 的系统函数。函数依字母顺序排列，分为以下部分：

概要（Summary） 简述程序作用，列出包含的文件，包括它的声明和原型，语法举例，和参数描述。

描述（Description）程序的详细描述，如何使用。

返回值 程序返回值说明。

参阅（see also） 相关程序。

例子 如何正确使用该函数的程序例子中断。

附注：

- 以 os_开头的函数可以由任务调用，但不能由中断服务程序调用。
- 以 isr_开头的函数可以由中断服务程序调用，但不能由任务调用。

1、isr_send_signal

概要： #include<rtx51tny.h>

char isr_send_signal(unsigned char task_id); /*信号发往的任务*/

描述： isr_send_signal 函数给任务 task_id 发送一个信号。如果指定的任务正在等待一个信号，则该函数使该任务就绪，但不启动它，信号存储在任务的信号标志中。

附注：

- 该函数是 RTX51 Tiny 实时操作系统的一部分，仅包含于 PK51 中。
- 该函数仅被中断函数调用。

返回值 成功调用后返回 0，如果指定任务不存在，则返回-1。

参阅 os_clear_signal,os_send_signal,os_wait

例子

```
#include<rtx51tny.h>
void tst_isr_send_signal(void) interrupt 2
{
   isr_send_signal(8); /*给任务 8 发信号*/
}
```

2、isr_set_ready

概要 #include< rtx51tny.h>

char isr_set_ready{ unsigned char task_id};/*使就绪的任务*/

描述 将由 task_id 指定的任务置为就绪态。

附注

- 该函数是 RTX51 Tiny 的一部分，包含在 PK51 中。
- 该函数仅用于中断函数。

返回值 无

例子 #include< rtx51tny.h>

```
void tst_isr_set_ready(void)interrupt 2
```

```
        { isr_set_ready(1);/*置位任务 1 的就绪标志*/
        }
```

3、os_clear_signal

概要　#include< rtx51tny.h>

```
      char os_clesr_signal(unsigned cahr task_id);/*清除信号的任
```

务*/

描述　清除由 task_id 指定的任务信号标志。

附注：该函数是 RTX51 Tiny 的一部分，包含在 PK51 中。

返回值　信号成功清除后返回 0，指定的任务不存在时返回-1。

参阅　isr_send_signal,os_send_signal,os_wait

例子　#include< rtx51tny.h>

```
    void tst_os_clsar_siganl(void)_task_8
    {
     …
     os_clear_signal(5);            /*清除任务 5 的信号标志*/
     …
    }
```

4、os_create_task

概要　#include<rtx51tny.h>

```
      char os_create_task(unsigned char task_id);/*要启动的任务
```

ID*/

描述　启动任务 task_id，该任务被标记为就绪，并在下一个时间点开始执行。

附注：　该函数是包含在 PK51 中的 RTX51 Tiny 的组成部分。

返回值　任务成功启动后返回 0，如果任务不能启动或任务已在运行，或没有以 task_id 定义的任务，返回-1。

参阅　os_delete_task

例子　#include< rtx51tny.h>

```
    #include<stdio.h>    /*用于 printf*/
    void new_task(void)_task_2
{…}
void tst_os_create_task(void)_task_0
{
    …
 if(os_create_task(2))
 {
  printf(“couldn’ t start task2\n”);
        }
            …
        }
```

5、os_delete_task

概要　#include<rtx51tny.h>

char os_delete_task(unsigned char task_id);/*要删除的任务*/

描述　函数将以 task_id 指定的任务停止，并从任务列表中将其删除。

附注　该函数是包含在 PK51 中的 RTX51 Tiny 的组成部分。

返回值　任务成功停止并删除后返回 0。指定任务不存在或未启动时返回-1。

附注　如果任务删除自己，将立即发生任务切换。

参阅　os_create_task

例子

```
#include<rtx51tny.h>
#include<stdio.h>
void tst_os_delete_task(void)_task_0
{
     …
  if(os_delete_task(2))
  {
     printf(“couldn’ t stop task2\n”);
  }
     …
}
```

6、os_reset_interval

概要

```
#include<rtx51tny.h>
void os_reset_interval(unsigned char ticks); /*滴答数*/
```

描述　用于纠正由于 os_wait 函数同时等待 K_IVL 和 K_SIG 事件而产生的时间问题，在这种情况下，如果一个信号事件（K_SIG）引起 os_wait 退出，时间间隔定时器并不调整，这样，会导致后续的 os_wait 调用（等待一个时间间隔）延迟的不是预期的时间周期。允许你将时间间隔定时器复位，这样，后续对 os_wait 的调用就会按预期的操作进行。

附注：　该函数是包含在 PK51 中的 RTX51 Tiny 的组成部分。

返回值　无

例子

```
#include<rtx51tny.h>
void task_func(void)_task_4
{
  …
  switch(os_wait2(KSIG|K_IVL,100))
     {
         case   TMO_EVENT:
      /*发生了超时，不需要 Os_reset_interval*/
               break;
         case    SIG_EVCENT:
         /*收到信号，需要 Os_reset_interval*/
              os_reset_interval(100);
         /*依信号执行的其它操作*/
```

```
                    break;
                }
            …
          }
```

7、os_running_task_id

概要　#include<rtx51tny.h>

```
char os_running_task_id(void);
```

描述　函数确认当前正在执行的任务的任务 ID。

附注：　该函数是包含在 PK51 中的 RTX51 Tiny 的组成部分。

返回值 返回当前正在执行的任务的任务号，该值为 0~15 之间的一个数。

例子　#include<rtx51tny.h>

```
void tst_os_running_task(void)_task_3
{
   unsigned char tid;
   tid=os_running_task_id( ); /*tid=3*/
 }
```

8、os_send_signal

概要　#include<rtx51tny.h>

```
char os_send_signal(char task_id);/*信号发往的任务*/
```

描述　函数向任务 task_id 发送一个信号。如果指定的任务已经在等待一个信号，则该函数使任务准备执行但不启动它。信号存储在任务的信号标志中。

附注　该函数是包含在 PK51 中的 RTX51 Tiny 的组成部分。

返回值 成功调用后返回 0，指定任务不存在时返回-1。

参阅　isr_send_signal,os_clear_signal,os_wait

```
#include<rtx51tny.h>
void signal_func(void)_task_2
{
   …
   os_send_signal(8);      /*向 8 号任务发信号*/
   …
 }
void tst_os_send_signal(void)_task_8
{
   …
   os_send_signal(2);     /*向 2 号任务发信号*/
   …
 }
```

9、 os_set_ready

概要　#include<rtx51tny.h>

```
char os_set_ready(unsigned char task_id);/*使就绪的任务*/
```

描述　将以 task_id 指定的任务置为就绪状态。

附注：　该函数是包含在 PK51 中的 RTX51 Tiny 的组成部分。

返回值　无

例子

```
#include<rtx51tny.h>
void ready_func(void)_task_2
{
    …
    os_set_ready(1);      /*置位任务 1 的就绪标志*/
    …
}
```

10、 os_switch_task

概要

```
#include<rtx51tny.h>
 char os_switch_task(void);
```

描述　该函数允许一个任务停止执行，并运行另一个任务。如果调用 os_switch_task 的任务是唯一的就绪任务，它将立即恢复运行。

附注：　该函数是包含在 PK51 中的 RTX51 Tiny 的组成部分。

返回值　无

例子

```
 #include<rtx51tny.h>
#include<stdio>
void long_job(void)_task_1
{
     float f1,f2;
     f1=0.0;
     while(1)
       {
          f2=log(f1);
          f1+=0.0001;
          os_switch_task();   /*运行其它任务*/
        }
  }
```

11、os_wait

概要

```
#include<rtx51tny.h>
char os_wait(
  unsigned char event_sel,     /*要等待的事件*/
  unsigned char ticks,        /*要等待的滴答数*/
  unsigned int   dammy);   /*无用参数*/
```

描述　该函数挂起当前任务，并等待一个或几个事件，如时间间隔，超时，或从其它任务和中断发来的信号。参数 event_set 指定要等待的事件，可以是下表中常数的一些组合。

事　件	描　　述
K_IVL	等待滴答值为单位的时间间隔
K_SIG	等待一个信号
K_TMO	等待一个以滴答值为单位的超时

事件可以用竖线符（“|”）进行逻辑或。例如，K_TMO1K_SIG 指定任务等待一个超时或者一个信号。

ticks 参数指定要等待的时间间隔事件(K_IVL)或超时事件(K_TMO)的定时器滴答数。参数是为了提供与兼容性而设置的，在中并不使用。

附注

- 该函数是包含在 PK 中的 RTX51 Tiny 的组成部分。
- 请参阅事件一节获得关于 K_IVL,K_SIG,K_TMO 的更多信息。

返回值 当有一个指定的事件发生时，任务进入就绪态。任务恢复执行时，下表列出的由返回的常数指出使任务重新启动的事件。可能的返回值有：

返 回 值	描　　述
RDY_EVENT	表示任务的就绪标志是被或函数置位的。
SIG_EVENT	收到一个信号
TMO_EVENT	超时完成，或时间间隔到
NOT_OK	参数的值无效

参阅

isr_send_signal,isr_set_ready,os_clear_signal,os_reset_interval,
os_send_signal,os_set_ready,os_wait1,os_wait2

例子

```
#include<rtx51tny.h>
#include<stdio.h>
void tst_os_wait(void)_task_9
{
   while(1)
    {
      char event;
      event=os_wait(K_SIG|K_TMO,50.0);
      switch(event)
      {
         default:                /*从不发生，该情况*/
         break;
         case    TMO_EVENT;/*超时*/
```

```
                    break;                  /*50 次滴答超时*/
                    case    SIG_EVENT; /*收到信号*/
                    break;
                  }
                }
              }
```

12、os_wait1

概要　#include<rtx51tny.h>

char os_wait1(unsigned char event_sel);/*要等待的事件*/

描述　该函数挂起当前的任务等待一个事件发生。os_wait1 是 os_wait 的一个子集，它不支持 os_wait 提供的全部事件。参数 event_sel 指定要等待的事件，该函数只能是 K_SIG。

附注：

- 该函数是包含于 PK51 中的 RTX51Tiny 的组成部分。
- 参见事件一节获得 K_IVL，K_SIG 和 K_TMO 的更多信息。

返回值　当指定的事件发生，任务进入就绪态。任务恢复运行时，os_wait1

返回的值表明启动任务的事件，返回值见下面的常数列表：

返　回　值	
RDY_EVENT	任务的就绪标志位是被 os_set_ready 或 isr_set_ready 置位的
SIG_EVENT	收到一个信号
NOT_OK	Event_sel 参数的值无效

例子　见 os_wait

13、os_wait2

概要　#include<rtx51tny.h>

```
char os_wait2(unsigned char event_sel,      /*要等待的事件
*/
                    unsigned char ticks);          /*要等待的滴答
数*/
```

描述 函数挂起当前任务等待一个或几个事件发生，如时间间隔，超时或一个从其它任务或中断来的信号。参数 event_sel 指定的事件可以是下列常数的组合：

Event	描　　述
K_IVL	等待以滴答数为单位的时间间隔
K_SIG	等待一个信号
K_TMO	等待以滴答数为单位的超时

事件可以用“|”进行逻辑或。如 K_TMO|K_SIG 表示任务等待一个超时或一个信号。
参数 ticks 指定等待时间间隔(K_IVL)或超时(K_TMO)事件时的滴答数。

附注：

- 该函数是包含于 PK51 中的 RTX51Tiny 的组成部分。
- 参见事件一节获得更多关于 K_IVL, K_TMO,和 K_SIG 的信息。

返回值　当一个或几个事件产生时,任务进入就绪态.任务恢复执行时, os_wait2 的返回值见下面的常数列表：

返 回 值	描　　述
RDY_EVENT	任务的就绪标志是被 os_set_ready 或 isr_set_ready 函数置位的
SIG_EVENT	收到一个信号
TMO_EVENT	返回时完成,或时间间隔到达
NOT_OK	参数 event_sel 的值无效

例子　　见 os_wait。