RTX51Tiny API 速查

目录

isr_send_signal....2

isr_set_ready....3

os_clear_signal....4

os_create_task....5

os_delete_task....6

os_reset_interval....7

os_running_task_id....8

os_send_signal....9

os_set_ready....10

os_switch_task....11

os_wait....12

os_wait1....14

os_wait2....15

说明....17

isr_send_signal

Summaty: char isr_send_signal (unsigned char *task_id*);

Description: **The isr_send_signal** function sends a signal to task *task_id.* If the specified task is already waiting for a signal,this function readies the task for execution but does not start it.Otherwise,the signal is stored in the signal flag of the task.

Note

A. This function is part of the RTX51 Tiny Real-Time Operating System which is included only with the PK51 Professional Developer's Kit.

B. This function may be called only from interrupt function.

Return Value: The **isr_send_signal** function returns a value of 0 if successful and -1 if the specified task does not exist.

Example:

```
#include <rtx51tny.h>
void tst_isr_send_signal (void) interrupt 2
{
isr_send_signal (8);          /* signal task #8 */
}
```

isr_set_ready

Summary: char isr_set_ready (unsigned char *task_id*);

Description: The **isr_set_ready** function places the task specified by *task_id* into the ready state.

Note

A. This function is part of the RTX51 Tiny Real-Time Operating System which is included only with the PK51 Professional Developer's Kit.

B. This function may be called only from interrupt function.

Return Value: None.

Example:

```
#include <rtx51tny.h>
void tst_isr_set_ready (void) interrupt 2
{
isr_set_ready (1);      /* Set ready flag for task 1 */
}
```

os_clear_signal

Summary: char os_clear_signal (unsigned char task_id);

Description: The **os_clear_signal** function clears the signal flag for the task specified by *task_id.*

Note

This function is part of the RTX51 Tiny Real-Time Operating System which is included only with the PK51 Professional Developer's Kit.

Return Value: The **os_clear_signal** function returns a value of 0 if the signal was successfully celared. A value of -1 is returned if the specified task does not exist.

Example:

```
#include <rtx51tny.h>
void tst_os_clear_signal (void) _task_ 8
{
os_clear_signal (5);    /* clear signal flag in task 5 */
}
```

os_create_task

Summary: char os_create_task (unsigned char *task_id*);

Description: The **os_create_task** function starts task number *task_id*. The task is marked as ready and begins execution at the next available opportunity.

Note

This function is part of the RTX51 Tiny Real-Time Operating System which is included only with the PK51 Professional Developer's Kit.

Return Value: The **os_create_task** function returns a value of 0 if the task was successfully started. A value of -1 is returned if the task could not be started,if the task is already running,or if no task was defined using the specified *task_id*.

Example:

```
#include <trx51tny.h>
#include <stdio.h>
void new_task (void) _task_ 2
{
…
}
void tst_os_create_task (void) _task_0
{
…
if (os_create_task (2))
{
    printf ("Couldn't start task 2\n");
}
}
```

os_delete_task

Summary: char os_delete_task (unsigned char *task_id*);

Description: The **os_delete_task** function stops the task specified by *task_id* and removes it from the task list.

Note

This function is part of the RTX51 Tiny Real-Time Operating System which is included only with the PK51 Professional Developer's Kit.

Return Value: The **os_delete_task** function returns a value of 0 if the task was successfully stopped and delete. A return value of -1 indicates the specified task does not exist or has not been started.

Note

A task switch is performed immediately if a task deletes itself.

Example:

```
#include <rtx51tny.h>
#include <stdio.h>
void tst_os_delete_task (void) _task_ 0
{
…
if (os_delete_task (2))
{
    printf ("Couldn't stop task 2\n");
}
}
```

os_reset_interval

Summary: void os_reset_interval (unsigned char ticks);

Description: The **os_reset_interval** function is used to correct timer problem that occur when the **os_wait** function is used to wait for **K_IVL** and **K_SIG** events simultaneously. In such a case, if a signal (**K_SIG**) events causes **os_wait** to exit, the interval timer is not adjusted and subsequent calls to **os_wait** (to wait for an interval) may not delay for the required time period.

The **os_reset_interval** routine allows you to reset the interal timer so that subsequent calls to **os_wait** operate as expected.

Return Value: None.

Example:

```
#include <rtx51tny.h>
void task_func (void) _task_ 4
{
switch (os_wait2 (K_SIG | K_IVL, 100))
{
case TMO_EVENT:
    /* Timeout occurred */
    /* os_reset_interval not required */
    break;
case SIG_EVENT:
    /* Signal received */
    /* os_reset_interval required */
    os_reset_interval (100);
    /* do something with the signal */
    break;
}
}
```

os_running_task_id

Summary: char os_running_task_id (void);

Description: The **os_running_task_id** function determines the *task_id* of the task currently executing.

Note

This function is part of the RTX51 Tiny Real-Time Operating System which is included only with the PK51 Professional Developer's Kit.

Return Value: The **os_running_task_id** function retuens the *task_id* of the task currently executing. This value is a number in the rang 0-15.

Example:

```
#include <rtx51tny.h>
void tst_os_running_task (void) _task_ 3
{
unsigned char tid;
tid = os_running_task_id ();     /* tid = 3 */
}
```

os_send_signal

Summary: char os_send_signal (char *task_id*);

Description: The **os_send_signal** function sends a signal to task *task_id*. If the specified task is already waiting for a signal, this function call readies the task for execution but does not start it. Otherwise, the signal is stored in the signal flag of the task.

Note

This function is part of the RTX51 Tiny Real-Time Operating System which is included only with the PK51 Professional Developer's Kit.

Return Value: The **os_send_signal** function returns a value of 0 if successfully and -1 if the specified task does not exist.

Example:

```
#include <rts51tny.h>
void signal_func (void) _task_ 2
{
…
os_send_signal (8);    /* signal task #8 */
…
}

void tst_os_send_signal (void) _task_ 8
{
…
os_send_signal (2);    /* signal task #2 */
…
}
```

os_set_ready

Summary: char os_set_ready (unsigned char *task_id*);

Description: The **os_set_ready** function places the task specified by *task_id* into the ready state.

Note

This function is part of the RTX51 Tiny Real-Time Operating System which is included only with the PK51 Professional Developer's Kit.

Return Value: None.

Example:

```
#include <rtx51tny.h>
{
…
os_set_ready (1);       /* Set ready flag for task #1 */
…
}
```

os_switch_task

Summary: char os_switch_task (void);

Description: The **os_awitch_task** function allows a task to halt execution and allow another task to run. If the task calling **os_swith_task** is the only ready for execution it resumes running immediately.

Note

This function is part of the RTX51 Tiny Real-Time Operating System which is included only with the PK51 Professional Developer's Kit.

Return Value: None.

Example:

```
#include <rtx51tny.h>
#include <stdio.h>
void long_job (void) _task_ 1
{
float f1, f2;
f1 = 0.0;
while (1)
{
    f2    = log (f1);
    f1 += 0.0001;
    os_switch_task ();      /* run other tasks */
}
}
```

os_wait

Summary: char os_wait (

unsigned char event_sel, /* events to wait for */

unsigned char ticks, /* timer ticks to wait */

unsigned int dummy); /* unused argument */

Description: The **os_wait** function halt the current task and wait for one or several events such as a time interval, a time-out, or a signal form another task or interrupt. The **event_sel** argument specifies the event or events to wait for and can be any combination of the following manifest constants:

Event	Description
K_IVL	Wait for the interval specified by **ticks.**
K_SIG	Wait for a signal
K_TMO	Wait for a time-out specified by **ticks.**

Events may be logically ORed using the vertical bar character ('|'). For example, **K_TMO | K_SIG**, specifies that wait for a time-out or a signal.

The **ticks** argument specifies the number of timer ticks to wait for an interval event (**K_IVL**) or a time-out event (**K_TMO**).

The **dummy** argument is provided for compatibility with RTX51 Full and is not used in RTX51 Tiny.

Note

A. This function is part of the RTX51 Tiny Real-Time Operating System which is included only with the PK51 Professional Developer's Kit.

B. Refer to Events for more information about **K_IVL, K_TMO,**

and **K_SIG.**

Return Value: When one of the specified events, the task is put in the READY state. When the task resumes execution, the manifest constant that identifies the event that restarted the task is returned by **os_wait.** Possible return values are:

Return Value	**Description**
RDY_EVENT	The task's ready flag was set by **os_set_ready** or **isr_set_ready.**
SIG_EVENT	A signal was received.
TMO_EVENT	A time-out has completed or an interval has expired.
NOT_OK	The value of the **event_sel** argument is invalid.

Example:

```
#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:          break;   /* this never happens */
    case TMO_EVENT:   break;   /* 50 ticks time-out */
    case SIG_EVENT:   break;   /* signal received */
    }
}
}
```

os_wait1

Summary: char os_wait1 (

unsigned char event_sel); /* events to wait for */

Description: The **os_wait1** function halts the current task and waits for an event to occur. The **os_wait1** function is a subset of the **os_wait** function and does not support all of the events that **os_wait** offers. The **event_sel** argument specifies the event to wait for and may have only the vlue **K_SIG** which waits for a signal.

Note

A. This function is part of the RTX51 Tiny Real-Time Operating System which is included only with the PK51 Professional Developer's Kit.

B. Refer to Events for more information about **K_IVL, K_TMO,** and **K_SIG.**

Return Value: When one of the specified events, the task is put in the READY state. When the task resumes execution, the manifest constant that identifies the event that restarted the task is returned by **os_wait1.** Possible return values are:

Return Value	**Description**
RDY_EVENT	The task's ready flag was set by **os_set_ready** or **isr_set_ready.**
SIG_EVENT	A signal was received.
NOT_OK	The value of the **event_sel** argument is invalid.

Example: See **os_wait.**

os_wait2

Summary: char os_wait2 (

unsigned char event_sel, /* events to wait for */

unsigned char ticks); /* timer ticks to wait */

Description: The **os_wait2** function halts the current task and waits for one or several events such as a time interval, a time-out, or a signal form another task or interrupt. The **event_sel** argument specifies the event or events to wait for and can be any combination of the following manifest constants:

Event	Description
K_IVL	Wait for the interval specified by **ticks.**
K_SIG	Wait for a signal
K_TMO	Wait for a time-out specified by **ticks.**

Events may be logically ORed using the vertical bar character ('|'). For example, **K_TMO | K_SIG**, specifies that wait for a time-out or a signal.

The **ticks** argument specifies the number of timer ticks to wait for an interval event (**K_IVL**) or a time-out event (**K_TMO**).

Note

A. This function is part of the RTX51 Tiny Real-Time Operating System which is included only with the PK51 Professional Developer's Kit.

B. Refer to Events for more information about **K_IVL, K_TMO,** and **K_SIG.**

Return Value: When one of the specified events, the task is put in the READY state. When the task resumes execution, the manifest constant that

identifies the event that restarted the task is returned by **os_wait2.** Possible return values are:

Return Value	Description
RDY_EVENT	The task's ready flag was set by **os_set_ready** or **isr_set_ready.**
SIG_EVENT	A signal was received.
TMO_EVENT	A time-out has completed or an interval has expired.
NOT_OK	The value of the **event_sel** argument is invalid.

Example: See **os_wait**.

说明

本文档参考 Keil 自带 rtx51tiny 手册

2013 年 1 月整理

版本 0.1

希望有人协助翻译成中文

By 若如初见

454636197@qq.com

2013.1.28 0:47