2019/12/3 zsy已阅

查看: 1318 | 回复: 46

开源个我的极简定时调度器 [复制链接]

tomzbj

1665 积分 | 67 主题 | 1531 帖子

金牌会员

发消息

发表于 5 天前 | 只看该作者

1楼 电梯直达

这是一个时间片轮转的多任务调度算法
每一个子任务都被周期性的调用，子任务必须不能阻塞，
并且以最短的时间运行完毕。

就俩文件，五个api：

ztask.h:

```
01.  #ifndef _ZTASK_H
02.  #define _ZTASK_H
03.
04.  #define ZT_MAX_TASKS 4
05.
06.  typedef void (*zt_func_t)(void);
07.
08.  // should be called in main loop
09.  void zt_poll(void);
10.
11.  // timeout: repeat inteval;   en: start immediately or not
12.  int zt_bind(zt_func_t func, int repeat, int en);
13.
14.  // should be called in systick_irqhandler
15.  void zt_tick(void);
16.
17.  void zt_start(int id);
18.  void zt_stop(int id);
19.
20.  #endif
21.
```

最大支持4个子任务

函数指针

复制代码

ztask.c:

```
01.  #include "ztask.h"
02.
03.  typedef struct {
04.      zt_func_t func;
05.      unsigned long timeout, repeat;
06.      int en;
07.  } zt_task_t;
08.
09.  static struct {
10.      unsigned long ticks;
11.      int num_tasks;
12.      zt_task_t tasks[ZT_MAX_TASKS];
13.  } g;
14.
15.  void zt_tick(void)
16.  {
17.      // overflow not handled. for 1ms tick, it could run 2^32 * 1ms = ~50 days.
18.      g.ticks++;
19.  }
20.
21.  void zt_poll(void)
```

全局，用于管理每一个子任务

时钟基准，每过1ms加1

```
{
    int i;
    for(i = 0; i < g.num_tasks; i++) {   依次遍历每一个子任务
        if(g.ticks >= g.tasks[i].timeout) {  如果当前时基超过了定义的时刻
            g.tasks[i].timeout = g.ticks + g.tasks[i].repeat;  重置超时为当前时基+repeat时间
            if(g.tasks[i].en)   如果子任务被使能，则调用子任务(被执行)
                g.tasks[i].func();
        }
    }
}

void zt_stop(int id)
{
    if(id < ZT_MAX_TASKS)
        g.tasks[id].en = 0;   设置使能标志位为0,则不会被调度
}

void zt_start(int id)
{
    if(id < ZT_MAX_TASKS)
        g.tasks[id].en = 1;  设置使能标志位为1,等待被poll调度
}

int zt_bind(zt_func_t func, int repeat, int en)
{
    if(g.num_tasks < ZT_MAX_TASKS) {
        g.tasks[g.num_tasks].func = func;
        g.tasks[g.num_tasks].repeat = repeat;
        g.tasks[g.num_tasks].timeout = 0;
        g.tasks[g.num_tasks].en = en;
        return g.num_tasks++;
    }
    else
        return -1;
}
```

复制代码

使用方法：
定义两个任务函数：

```
void blink(void)    子任务必须不能阻塞，要不整个调度器也就被阻塞了
{
    GPIOA->ODR |= GPIO_PIN_1;
    _delay_us(10);
    GPIOA->ODR &= ~GPIO_PIN_1;
}

void hello(void)   子任务必须不能阻塞，要不整个调度器也就被阻塞了
{
    printf("Hello, world.\n");
}
```

复制代码

在进入主循环前：

```
    zt_bind(blink, 100, 1);      // 每100ms执行一次
    zt_bind(hello, 500, 1);     // 每500ms执行一次
```

复制代码

最后在主循环里调用 zt_poll(), 在systick中断里调用zt_tick()， 就行了。

如果需要控制任务启动/停止， 需要用一个变量保存zt_bind的返回值，然后执行zt_start和zt_stop即可。