OpenHarmony内核开发—定时器管理

前言

本节主要介绍：

- 定时器的相关概念
- 定时器的运作机制
- 如何启动和关闭定时器

目录

软件定时器基本概念

软件定时器，是基于系统Tick时钟中断且由软件来模拟的定时器，当经过设定的Tick时钟计数值后会触发用户定义的回调函数。定时精度与系统Tick时钟的周期有关。

硬件定时器受硬件的限制，数量上不足以满足用户的实际需求，因此为了满足用户需求，提供更多的定时器，LiteOS操作系统提供软件定时器功能。

软件定时器扩展了定时器的数量，允许创建更多的定时业务。

软件定时器功能上支持：

- 静态裁剪：能通过宏关闭软件定时器功能。
- 软件定时器创建。
- 软件定时器启动。
- 软件定时器停止。
- 软件定时器删除。
- 软件定时器剩余Tick数获取。

软件定时器运作机制

软件定时器使用了系统的一个队列和一个任务资源，软件定时器的触发遵循队列规则，先进先出。定时时间短的定时器总是比定时时间长的靠近队列头，满足优先被触发的准则。

软件定时器以Tick为基本计时单位，当用户创建并启动一个软件定时器时，Huawei LiteOS会根据当前系统Tick时间及用户设置的定时间隔确定该定时器的到期Tick时间，并将该定时器控制结构挂入计时全局链表。

当Tick中断到来时，在Tick中断处理函数中扫描软件定时器的计时全局链表，看是否有定时器超时，若有则将超时的定时器记录下来。

Tick中断处理函数结束后，软件定时器任务（优先级为最高）被唤醒，在该任务中调用之前记录下来的定时器的超时回调函数。

实现软件定时器创建

cmsis_os2的API软件定时器接口简介：

接口名	功能描述
osTimerNew	创建定时器
osTimerStart	启动定时器
osTimerStop	停止定时器
osTimerDelete	删除定时器

创建定时器： osTimerNew (osTimerFunc_t func, osTimerType_t type, void *argument, const osTimerAttr_t *attr);

启动定时器： osTimerStart (osTimerId_t timer_id, uint32_t ticks);

停止定时器： osTimerStop (osTimerId_t timer_id);

删除定时器： osTimerDelete (osTimerId_t timer_id);

软件定时器扩展实验

扩展实验代码

```
exec1 = 1U;
id1 = osTimerNew(Timer1_Callback, osTimerOnce, &exec1, NULL);
if (id1 != NULL)
{

    // Hi3861 1U=10ms,100U=1S
    timerDelay = 100U;

    status = osTimerStart(id1, timerDelay);
    if (status != osOK)
    {
        // Timer could not be started
    }
}
```

```
osDelay(50U);
status = osTimerStop(id1);
if(status != osOK)
{
    printf("stop Timer1 failed\r\n");
}
else
{
    printf("stop Timer1 success\r\n");
}
status = osTimerStart(id1, timerDelay);
if (status != osOK)
{
    printf("start Timer1 failed\r\n");
}
osDelay(200U);
status = osTimerDelete(id1);
if(status != osOK)
{
    printf("delete Timer1 failed\r\n");
}
else
{
    printf("delete Timer1 success\r\n");
}
```

本节小结

- 1、了解软件定时器的概念
- 2、掌握如何创建并启动定时器
- 3、掌握如何停止和删除定时器

谢谢观看

开 源 从 小 熊 派 开 始

OPEN-SOURCE STARTED WITH THE BEARPI