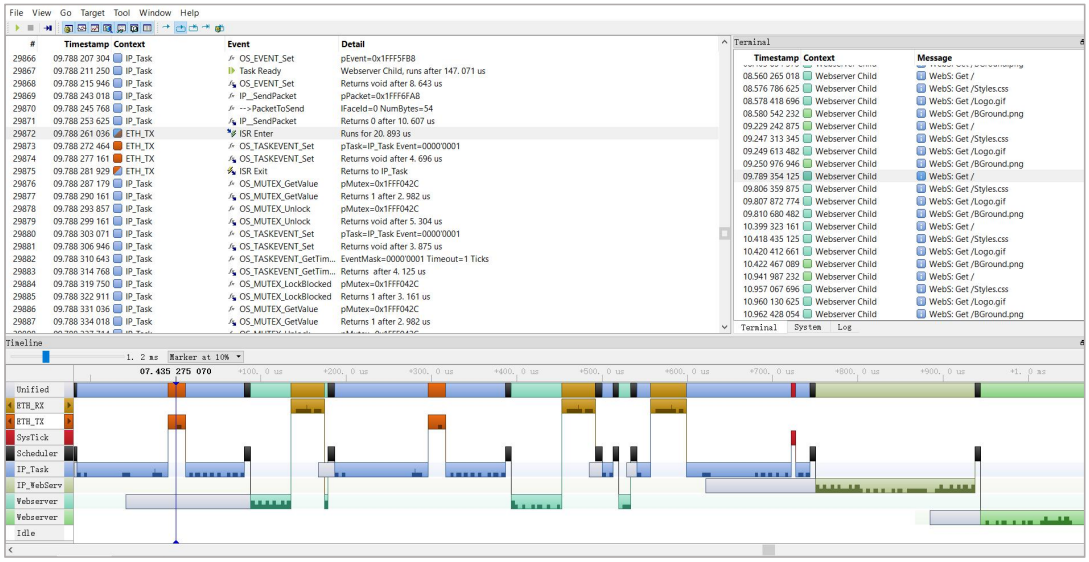
SystemView 3.10 分析 FreeRTOS 时出现闪退的原因

SystemView 是 SEGGER 公司推出的一款运行时可视化分析工具，支持 FreeRTOS、μC/OS、embOS 等 RTOS，及无 RTOS 的系统。需要使用 J-Link 调试器，处理器芯片需支持内存后台访问(允许 CPU 在运行时通过调试端口访问内存)，SystemView 使用起来非常方便。

2019 年 12 月的时候，SEGGER 发布了 SystemView 3.10 版本。该版本做了比较大的升级，取消了非授权版本 100 万事件和过滤器使用的限制，评估版本与商业授权版本在功能上完全一样，只需在启动软件时声明自己用于非商业用途。此外，3.10 版本在可视化效果和操作细节上也有改进，用户也可以对窗口进行自定义布局。



但是 3.10 版本在 FreeRTOS 系统上使用时，开启记录后可能会出现 SystemView PC 端软件闪退的问题，而使用 3.10 之前的版本，不会出现这个情况。经过测试发现， 对于 3.10 的版本，如果没有对 FreeRTOS 调度器进行跟踪，就会出现闪退的问题。

解决的办法是，在 FreeRTOS 的 port.c 中的 xPortSysTickHandler()函数内加上对中断和调度器的跟踪。

```
void xPortSysTickHandler( void )
{
#if (defined(config USE_TRACE_FACILITY) && config USE_TRACE_FACILITY > 0)
    traceISR_ENTER();
#endif

    /* The SysTick runs at the lowest interrupt priority, so when this interrupt
    executes all interrupts must be unmasked.  There is therefore no need to
    save and then restore the interrupt mask value as its value is already
    known. */
    portDISABLE_INTERRUPTS();
    {
        /* Increment the RTOS tick. */
        if( xTaskIncrementTick() != pdFALSE )
        {
#if (defined(configUSE_TRACE_FACILITY) && configUSE_TRACE_FACILITY > 0)
            traceISR_EXIT_TO_SCHEDULER();
#endif
            /* A context switch is required.  Context switching is performed in
            the PendSV interrupt.  Pend the PendSV interrupt. */
            portNVIC_INT_CTRL_REG = portNVIC_PENDSVSET_BIT;
        }
        else
        {
#if (defined(configUSE_TRACE_FACILITY) && configUSE_TRACE_FACILITY > 0)
            traceISR_EXIT();
#endif

        }

    }
    portENABLE_INTERRUPTS();
}
```

其中，traceISR_ENTER()和 traceISR_EXIT()用于跟踪 ISR 的进入和退出，以此来获得 ISR 的执行时间信息，对于要进行跟踪的每个中断，需要在 ISR 的开头调用 traceISR_ENTER()，并在在最后调用 traceISR_EXIT()。traceISR_EXIT_TO_SCHEDULER()表示 FreeRTOS 退出时钟节拍 ISR 后，要进行任务调度。

如果在代码中没有按照上述方法对系统节拍中断和调度器进行跟踪，SystemView 3.10 的应用程序就会卡死闪退。这个问题我们已经反馈给了 SEGGER，在后面的版本会进行改进。

SystemView 对于要学习 RTOS 原理的人员来说是非常理想的工具，有关 SystemView 的使用和下载可以访问 SEGGER 官网：https://www.segger.com/。