Purdue University Fort Wayne

ECE 485

Lab 2: RTOS Kernel

Yilei (Eddie) Li, Rishi Mitra

10/11/2021

1.0 Objective

The objective of the lab is to use the TI TM4C123G LaunchPad board, µVision development system and the TM4C123 ARM Cortex-M4 microcontroller in order to:

- Develop OS facilities for real-time applications,
- Coordinate multiple foreground and background threads,
- Design a round robin multi-thread scheduler
- Learn to use the Tiva Firmware library in our designs

2.0 Software Design

2.1 Cooperative scheduling

Cooperative scheduling is a style of scheduling in which the OS never interrupts a running process to initiate a context switch from one process to another. Processes must voluntarily yield control periodically or when logically blocked on a resource.

2.2 Round robin scheduling

Round Robin is the preemptive process scheduling algorithm implemented in this part. Each process is provided a fix time to execute. Once a process is executed for a given time period, it is preempted and other process executes for a given time period. Context switching is used to save states of preempted processes.

2.3 Implementing Lab 1 using the TivaWare C Series software suite

Lab 1 involved setting up of ADC one-shot and ADC buffer collection. This was then implemented in a command-driven interface using PuTTY which communicated with a UART driver that was designed to accept commands, execute tasks and print the results to the ST7735 LCD.

3.0 Analysis and Discussions

1) **Why did the time jitter in my solution jump from 4 to 6 µs when interpreter I/O occurred?**

   The jump of the timer jitter is caused by the interference cause by scheduling of tasks and handling of asynchronous events (interrupts). In this lab, multiple tasks were scheduled using cooperative and round robin scheduling and interrupts was enabled, which has resulted in the jitter jump when the interpreter I/O occurred.

2) **Explain why the LED patterns behave differently for cooperative scheduling, round-robin operating, and round-robin with different number of time slices.**

Different scheduling methods allocate execution time differently for each task. This results in different LED patterns emerging(which can be viewed in the images below)

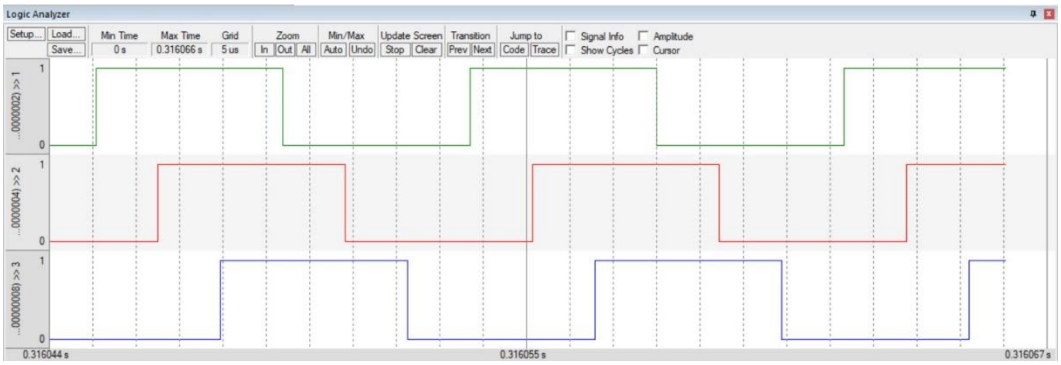

Figure 1. Logic analyzer profiling PF1, PF2, and PF3 for cooperative thread switching.

With cooperative scheduling, the tasks themselves are written in such a way that they cooperate with each other, yielding execution when necessary. As we can see from the figure that task 1 begin running, triggering task 2 to run, which in turn triggers task 3. The overlapping period resulted in different LED colors. The pattern can be seen in the table below.

Table 1. LED color pattern

PF1: Red	ON	ON	ON	ON	ON	ON	OFF	OFF	OFF	OFF	OFF	OFF	ON	ON
PF2: Blue	OFF	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF	OFF	OFF	OFF	OFF
PF3: Green	OFF	OFF	OFF	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF	OFF	OFF
Resulting color	red	red	red+blue	red+blue	white	white	light blue	light blue	green	green	no light	no light	red	red

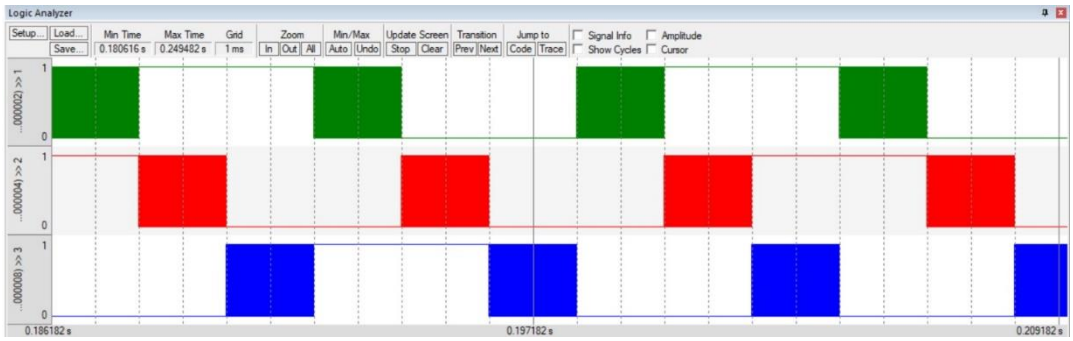

Figure 2. Logic analyzer profiling for Round-Robin

In Round robin scheduling, the execution of tasks can be seen in a circularly-sequential manner. Task 1 was allowed to execute (turning red LED on and off) for a fix slices, following by task 2 and 3 executed in the same manner. The length of the time slice defines how long a thread will execute before a thread switch, resulting in different LED patterns.