# Lab 3 RTOS with FIFO, blocking, and sleeping

**Goals**  • Extend the RTOS to include priority scheduling.

* Extend the RTOS to include a hardware interrupt on button push.

# Starter files • Lab 4 files in tm4c123valvanoware folder.

• Your Lab 3 solution will be helpful as well.

**Background**

In this lab you will implement a priority scheduler to allow more important threads to run more often than less important ones. You will need to update you TBC, scheduler, and initialization to make this happen. You will also implement a hardware interrupt using a button found on the boosterpack board.

## Preparation

1. Review the Real Time Systems chapter (5) of your book for background in how to implement the features of this lab as instructed.
2. Go through the os.c from Lab 3 and the os.c for Lab 4 and copy C code from Lab 2 to Lab 4 (do not move the entire file, just some C functions). Similarly, copy the SysTick ISR from Lab 4 osasm.s to your Lab 4 osasm.s. The Lab 3 SysTick ISR should be sufficient for Lab 4.

## Procedure

1. To get main\_step1 to run you will need to extend **OS\_AddThreads** to handle 8 threads, this should be simple and as you have done it before in other labs. You will also need to add a priority field to the TCB and implement that into **OS\_AddThreads** as well. To accommodate the priority field you should adjust your scheduler to a priority scheduler instead of a round-robin scheduler. Be sure to read section 5.2 to learn how priority schedulers work.
2. In this step you have to implement the functions **OS\_PeriodicTrigger0\_Init** and **OS\_PeriodicTrigger1\_Init** to create periodic tasks in place of the last system we implemented to run periodic tasks.
3. In this step you will need to implement **OS\_EdgeTrigger\_Init**, **OS\_EgdeTrigger\_Restart**, and **GPIOPortD\_Handler** to trigger and go to the next screen when the button on the booster pack is pressed using edgeSemaphore. In the UH class we have decided to use button J4.32 instead of J4.33, this is to test that you can read the booster pack, find the correct pin address on the microcontroller, and activate it correctly using instructions for a different button.
4. Step 6 is the final step in which you test the actual main function and run the full lab to see if your operating system works.

## Checkout (show this to the TA)

1. Demonstrate the final system to the TA.

## Deliverables (exact components of the lab report)

1. Each lab member should submit a copy of the code to the lab 4 assignment under assignments on blackboard. Preferred submittal is in the form of a zip/rar/7z file.

## Hints

1. You should go through the procedure section in order running main\_step1 with step 1, main\_step2 with step 2 and so on until step 6 when you can run the actual main function. Be sure to change the names of each main function as you can only have one at a time.
2. Since each of the labs is built on top of the previous lab, time spent debugging this lab will greatly simplify subsequent labs. In other words, some students report that significant time is wasted during labs 4 5 and 6 because their Lab 3 OS has bugs.
3. Be sure to review how pointers in C work as their misuse can cause a lot of issues that can take a lot of time to sort out.