Tutorial 3: Use of the Zephyr RTOS for Embedded Micro-controller-based RT Applications

42081 – Real-Time and Operative Systems

Team

- David José Araújo Ferreira - 93444
- Tiago Adónis Hipólito Dias - 88896

Tasks

For this tutorial, three main task needed to be configured in the program:

- a **control for the LEDs**,
- a set of **three tasks** using threads, shared memory and semaphores;
- a controller to **receive the signal** line from the potentiometer.

Functionalities

The control of the **blinking LEDs** is done by the *main* method, where the LEDs will all blink when the global variable *blink* is **1 or greater**, set by pressing the **button 1**.

For the tasks of **sampling the signal**, **calculating the moving average** and **outputting** the values to the console and LEDs, it was created:

- **3** threads: *sampler*, *filter*, *output*;
- **2** semaphores: *ab* and *bc*;
- **3** shared variables: *new_sample*, *filtered*, *blink*.

Sequence of events

1. For sampling, the ADC is read by the *sampler* task, which stores this value in the *new_sample* buffer and *give* the semaphore *sem_ab*. It will then sleep for a designated sleeping period, after which will **repeat the process**.
2. The task *filter*, after creation, will **wait for *sem_ab*** to activate. This task's method contains an array that, in a circular fashion,** stores the last 10 values passed** in the *new_sample* buffer. After each calculation of a moving average, the task *filter* will store this calculated value in *filtered* and then **give the semaphore *sem_bc***.
3. The *output* task, after creation, **will wait for *sem_bc*** to activate. When the semaphore is taken, this task's method will **check the variable *blink*** which will be **different** than **0** if the blinking process is taking place, and if it is not, it will then set the LEDs for the corresponding states according to the value of *filtered*.