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

Introduction

The aim of this tutorial is to learn how to implement a real-time application composed of a set of real-time tasks using the Zephyr RTOS in a micro-controller environment.

Following the typical structure of embedded software, a mix of periodic and sporadic tasks will be considered.

Preparation

Students should first get comfortable with the Nordic NCS toolchain and with the development kit. To this end, they should analyze the supplementary materials provided in the course website (including e.g. introduction to Cmake, Devicetrees and Zephyr) and study in detail all the examples that are provided. Each example addresses a specific topic that is relevant for the SOTR course.

Specification

The system to implement emulates a distance detector system. The distance sensor is emulated by a potentiometer (important details on this are presented below) that generates at its output an analog voltage comprised between 0 and 3. The sensitivity of the sensor is 100 mV/m. The output is carried out by the devkit leds, as indicated in the following table:

Distance (m)	Leds ON
>= 30	Led 1
[20, 30[	Leds 1,2
[10, 20[	Leds 1,2,3
[0,10[	Led 1,2,3 and 4

The distance sensor is assumed to be noisy, so its output must be filtered before use. It should be employed a moving average filter, with a window size of 10 samples.

Button “BUT1” is a test button. When pressed, the system should blink all leds during 5 seconds, stopping the normal execution of the software during this interval.

The system shall be structured with at least three tasks, matching the basic processing blocks, namely one task for acquiring the sensor samples, one for filtering and the other to output the signal (i.e. set the leds on/off).

The sampling task is periodic, while the other two are sporadic, being activated when new data is available. The sampling period, in milliseconds, is specified in a macro called "SAMP_PERIOD_MS".

The inter-task communication should be based on Shared Memory + Semaphores

As sated above, the input sensor is emulated by a potentiometer. It should be used a 10 kΩ potentiometer, supplied by the DevKit 3 V supply (VDD). **Make sure you use VDD (3V) and not the 5 V supply. If you use 5 V you can damage the microcontroller.**

Deliverables

- A two page report, pdf format, submitted via eLearning, with:
  - Page 1:
    - Identification of the course and assignment
    - Identification of the group members (first name, last name and ID number)
  - Page 2:
    - Brief description (diagrams and text) of the **task execution sequence and relevant events** when the system is working normally (not in test mode)
      - The diagram should contain the start/finish times of tasks, operations on the buffers, signals exchanged, etc. A reader should be able to understand which task does what and when.
- Zip file with the full project folder and files.
  - **The project code MUST be buildable from a fresh NCS installation!** Be careful to not include files that are outside of the project folder. Failing to comply with this rule implies that the work will not be evaluated, thus it will be assigned with a mark of 0 (zero)!