ET 095G Programming Exercise 6: File Management

September 6, 2021

1 Background and Learning Goals

In embedded sensor systems, data often has to be backed up for future use. For this it is essential that data is safely stored, even when power is lost. In this exercise you will learn how to read and write files on local file storage. After this exercise, you should be able to:

- create files on the mbed's local flash drive.
- write data to a file.
- read data from a file.

2 Preparation

Required information for this exercise can be found in chapter 10 of the course book. You can find further information on mbed and its API functions at https://docs.mbed.com. You might also need to get some additional information on file handling in C available online.

3 Evaluation and Presentation

For evaluation, submit a functioning .bin file and the main.cpp file of your program for each of the subtasks. The main file should be well commented. For submission, use the respective inbox for this exercise on the moodle course page.

4 Task

Write a program that stores temperature readings on the mbed's local flash memory. Every time the user presses the button (joystick center), five temperature values should be written in a new line to a file (add some delay between readings to allow the temperature to change). The five values should be written in one line, each value separated with a comma. The values should also be human-readable and should not overwrite existing values in the file. Once the values have been written, the program should read the last line (i.e., the same five values that have just been written) from the file and output it on a terminal emulator. Verify that you can read the values in a text editor, even after power has been cycled.

Note: There are different approaches to read the last line of a file. The 'fseek' function allows you to place the cursor in your file and can be helpful for this task. Note: It is good practice to try to keep ISRs as short as possible. In this application, for example, it is not recommended to perform everything that should happen on a button press within the button ISR.