Operating Systems Project

File System (FS) – Outline

Overview

In this project, you will develop a simple File System.

We assume that we have a physical medium with following properties:

- 20 Sectors, 64KB each
- Erase operation can be performed either on a single sector or on the whole physical medium (i.e., all 20 sectors). Erase operation sets all **bits** of the sector(s) to 1.
- Write operation can be performed only in words and on a word boundary (word is defined as 2 bytes). Write operation can set bits to 0, but **cannot** set bits to 1, i.e., it is equal to the AND operation between the word on the "physical medium" and the word which should be written.
- Addressing is "flat" beginning with the 0 byte of the first sector

Program requirements

- Programming languages: C/C++ only
- Compilation: Windows OS / MS Visual Studio
- Execution: Windows OS

Part 1

In first project, following "driver" operations (functions) should be implemented

- EraseSector (int nSectorNr)
- EraseAllSectors ()
- ReadWord (int nAddress)
- WriteWord (int nAddress, nWord)

It is your responsibility to decide whether these functions are void or return value, and which one.

The "physical medium" is simulated using a binary file which you should create (e.g., if any of the "driver" functions is called and file is not present).

Part 2

Design how you would manage files on such medium. Only single directory (root) is required, i.e., **no** directory management is needed.

Files should be uniquely identified by a variable-length file names (length 1 to 64 bytes / ASCII characters).

Specify:

- How information about files and their location should be stored on the "physical medium"
- What happens when file is created
- How file system can support changes of file content
- How file can be deleted

Following questions should be answered:

- Advantages and Disadvantages of the proposed solution
- How multiple open files can be managed
- How robust is the proposal in the case of failure during one of FS operations

Part 3

Using functions of "driver" layer, implement following functions:

- COMP3500_fopen
- COMP3500_flocse
- COMP3500_fread
- COMP3500_fwrite
- COMP3500_fseek
- COMP3500_ftell
- COMP3500_remove

All these functions should be an equivalent of fopen, fclose, fread, fwrite, etc. functions (see MSDN documentation for the function format and parameter meaning).

Attention: in opposite to the "driver" layer functions, these functions should be able to read and write on the **byte** boundary.

In order to evaluate FS, create, read, and write several files using these functions.

Part 4 (BONUS, i.e., not mandatory)

Implement Garbage Collector, which consolidates space used by deleted files and enables its usage.