Sam Gass, Haden Stuart, Haoqian Li, Henry Paek, Marvin Bell, Edward Ojo

Operating Systems Project Part 2

**Structures:**

**File Allocation Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Filename | Sector # | Offset | Size |
| Size | 64 bytes | 2 byte | 16 bytes | 16 bytes |
| Description | Unique string representing name of file | Which sector the file is located in | Number of words from start of sector | The number of bytes in the file |

The File Allocation Table (FAT) is a table of entries for the location of all files in memory. The FAT takes up the entire first sector of the memory. Each entry consists of the field shown in the table above. Each file has a unique string to identify it as a filename. That file is stored in the specified sector, at the specified offset from the start of that sector, and its content will be present in the next n bytes specified the size field.

**Free Space Table**

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Sector # | Offset | Size |
| Size | 2 byte | 16 bytes | 16 bytes |
| Description | Which sector the free space is located in | Number of words from start of sector | The number of free bytes at this location |

The Free Space Table (FST) is a table of entries for the location of all the free space where file can be written. It is stored at the end of the last sector in memory and added to backward. Each entry specifies the sector the free space is in, its offset from the start of that sector, and how many bytes are free after that offset. At the beginning of execution, the free space table will have 19 entries, one for each sector except for the sector with the FAT.

**Open File Table**

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Sector # | Offset | Size |
| Size | 2 byte | 16 bytes | 16 bytes |
| Description | Which sector the file is located in | Number of words from start of sector | The number of bytes in the file |

The Open File Table is a table of entries for the location of all open files. It is stored in memory. It is implemented as a queue and a first in first out implementation is used to manage file priority in memory.

**How this FS works:**

**When a file is created**

1. The FST is checked to location the first free space (Continuous Allocation, first fit)
2. Writes file into that sector at that offset
3. A new entry is added to the FAT for that file
4. That free space is removed from the FST

**When a file is modified**

1. Its location is found from the FAT
2. Seek to that file location
3. Seek to the part of the file to be overwritten
4. Overwrite that part of the file

**When a file is deleted**

1. Its entry is deleted from the FAT
2. That free space is added to the FST

**Advantages of this implementation**

1. Most parts are easy to implement
2. No external fragmentation between sectors
3. All parts of a file are quickly located

**Disadvantages of this implementation**

1. Bad performance for many functions
2. Internal fragmentation within sectors
3. Lots of wasted space on tables.
4. Files can be modified but their size cannot be changed without creating new file