Assignment 6 | File System
Arun Nekkalapudi [anekkal]

In Assignment 6 we have implemented in memory file system from scratch.

In fs.h we have five important structures [filetable (to store the file details like state,flieptr and pointer to dirent and inode structure),dirent(To Store the directory entries),directory(To store the directory details),fsystem (To store the file system details),inode] which together with other flags defines the top level functionalities of File System implementation.

I have implemented 6 important functions in this Assignment to support the implementation of file system.

1) fs_open()

   - **ARGUMENTS**: **filename, flags**
   - **DESCRIPTION:** In this function we will iterate through all the directory entries. If found, we will change the fileFound flag from 0 to 1 and returns file descriptor which is an integer.

2) fs_close()

   - **ARGUMENTS**: **fileDescriptor.**
   - **DESCRIPTION:** Check if the file exists or not If yes change the state of the file to closed. If not found, throw System Error and return or else return OK on Success.

3) fs_create()

   - **ARGUMENTS**: **filename, flags.**
   - **DESCRIPTION:** first check if the file exists if the file with same name exists throw an error and also check if there are any other conflicts associated with the file and throw error if there any, if not carry out the operations such as creating an entry in the file table and update all the structure that are associated with it.
     Upon successfully assigning the values, return file descriptor.

4) fs_seek()

   - **ARGUMENTS: fileDescriptor, offset**
   - **DESCRIPTION:** updating the offset based on the new file pointer value. If the offset value exceeds the file boundary we have set, then reset the offset to 0. And returns file descriptor.

5) fs_read()

   - **ARGUMENTS: fileDescriptor, buffer, nbytes**
   - **DESCRIPTION:** Based on the fileDescriptor get the inode and find out the blocks that are associated with the fileDescriptor [which will be certain set of blocks].After fetching all the blocks read the data from the blocks by copying required number of files to the output buffer.

6) fs_write()

   - **ARGUMENTS: fileDescriptor, buffer, nbytes**
   - **DESCRIPTION:** Get an inode for the file descriptor and write data into the blocks as directed by the structure inode. Update the file pointer and return the size of the file.

Lessons Learned:

- Clear understanding on how to implement simple File System on Xinu Operating System.
- Understood how we can store content in the form of blocks in memory using memcpy().

Github checkin URL:

https://github.iu.edu/anekkal/OS-P536-F17/commit/da1e49b0f4ea89d65bbafdc74a13e00e5b0a21d1