write_cp.c

int write_file (char *pathname)

First, I ask for an fd and text string to be written. I calculate number of bytes of the user given text string and return mywrite (fd, txt, nbytes).

int mywrite (int fd, char *buf, int nbytes)

First, I allocate an openFileTable entry and point it at the running->fd[fd] and check it's open for a valid writing mode. I point a MINODE at the oftp->mptr and record total number of bytes to be read as (nb). Then, I run a while loop, (while nbytes > 0) to write until nothing is left to be written. Inside, I calculate logical block (lbk) and starting byte. Then I use mailmans algorithm to convert lbk into a physical block number. (I also check to see if I need to allocate data blocks for direct, indirect, and double indirect data blocks) I get the physical block contents into a buffer (wbuf) and set a pointer to the startbyte of wbuf (Also calculate remaining # of bytes in block). I calculate the max number of bytes to be written in one operation (the minimum of remaining and nbytes). Next, I call memcpy to copy max number of bytes from wbuf into the buffer (char *buf parameter). I decrement remaining number of bytes and nbytes number both by the number of bytes written (max). I increment the offset by max, and if nelessary increase the size of the file by max as well. I write the wbuf back to the disk and loop back out until all bytes are written (nbytes ==0). Finally, I mark the MINODE as dirty and return (nb) total number of bytes read.

int cp (char *src, char *dest)

First, I (a)ll open_file for read mode to open the src file to be read from and I call open_file for write mode to open the dest file to be written to. I run a loop while reading from src into a buf, I write to dest from that buf the number of bytes read.

int mv (char *src, char *dest)

I begin by calling getino to get src inode number and make sure the file exists. Then I load the file into a MINODE and check to make sure the src file is on the current device. If src is on the current dev, I call link (src, dest) to hard link the dest to the src file (same INODE number), then unlink (src) to remove src name from the parent directory & dec. link count. If src is not on the current device, I call cp(src, dest) to copy the contents of src into the dest file. Then I call unlink (src) to remove src from parent DIR and decrease links count by 1.