Implementation of an extent based file system required the following changes:

Header Files

`fcntl.h`—Defined `O_EXTENT` as 0x201

`fs.h`—Defined `MAXEXTENT` as 6*256 to ensure the extent file didn't exceed 2^8 blocks

`stat.h`—Defined `T_EXTENT` for the new file type & added address and length fields to the stat structure

Source Files

`fs.c`—Updated `bmap()` to allocate adjacent disk data blocks and keep track of the length if the inode type was `T_EXTENT`. Maintained backward compatibility with `T_FILE` types. See code comments.

`fs.c`—Updated `itrunc()` to discard contents of unused inodes of type `T_EXTENT` when dereferenced.

`ls.c`—Updated to handle `T_EXTENT` files the same way it handles `T_FILE`

`sysfile.c`—Updated `create()` to handle `T_EXTENT` files like `T_FILE`

`sysfile.c`—Updated `sys_open()` to create an `EXTENT` inode if the `O_EXTENT` flag was used

Implementation of `fstat()` to include `EXTENT` file info required the following changes:

Header Files

`stat.h`—Added address and length fields to the stat structure

Source Files

`fs.c`—Updated `stati()` to populate the stat structure address and length fields

Implementation of `lseek()` system call required the following changes:

Header Files

`syscall.h`—Defined `SYS_lseek` as system call 22

`user.h`—Added `lseek()` definition for user programs to call

`usys.S`—Added `lseek` to system call handler

Source Files

`syscall.c`—Added `lseek` to system call array and extern list

`sysfile.c`—Implemented `sys_lseek()` which retrieves the file struct referenced by the fd and updates the offset