**5c. Implementing the Fast File System**

To building a new mkfs tool for the main fast file system optimizations involves several steps. The following is the overview of the process:

1. Introduce block groups:

Modify the file system layout to include block groups. Each block group should contain a certain number of data blocks, a group descriptor, and a backup of the superblock. Update the mkfs tool to create the new file system layout with block groups.

1. Add a new block allocation policy based on directory boundaries:

Modify the block allocation algorithm in the file system code to allocate blocks based on directory boundaries. Allocate new blocks for a file in the same block group as the parent directory, if possible. Update the mkfs tool to account for this new policy.

1. Properly handle block allocation for large files:

Implement a new block allocation policy for large files that may span multiple block groups. You may want to use indirect or double-indirect blocks to reference data blocks in other block groups. Update the mkfs tool to create the necessary data structures for large files.

1. Add new system calls and user programs:

Implement new system calls and user programs that can display block allocation information and demonstrate the optimized file system handling different file structures. You may need to create system calls for retrieving block allocation information and test user programs that create various file structures (e.g., small files, large files, deep directory hierarchies).

1. Testing:

Thoroughly test the new file system implementation by creating and manipulating files and directories using the modified mkfs tool, the new system calls, and the test user programs. Make sure that the file system handles various file structures as expected and that the block allocation policies are working correctly.

Here's a general guideline on how to proceed:

1. Update the mkfs tool to create the new file system layout with block groups. Modify the file system data structures as needed.
2. Modify the file system code in the kernel to implement the new block allocation policies and handle large files.
3. Implement the new system calls and user programs to demonstrate the optimized file system.
4. Testing the implementation thoroughly to ensure correct behavior.