Project 3 Readme

Group Members:

Prati Jain

Aditya Khetarpal

Rajdeep Savani

1. Adding lseek support

Implementation of lseek():

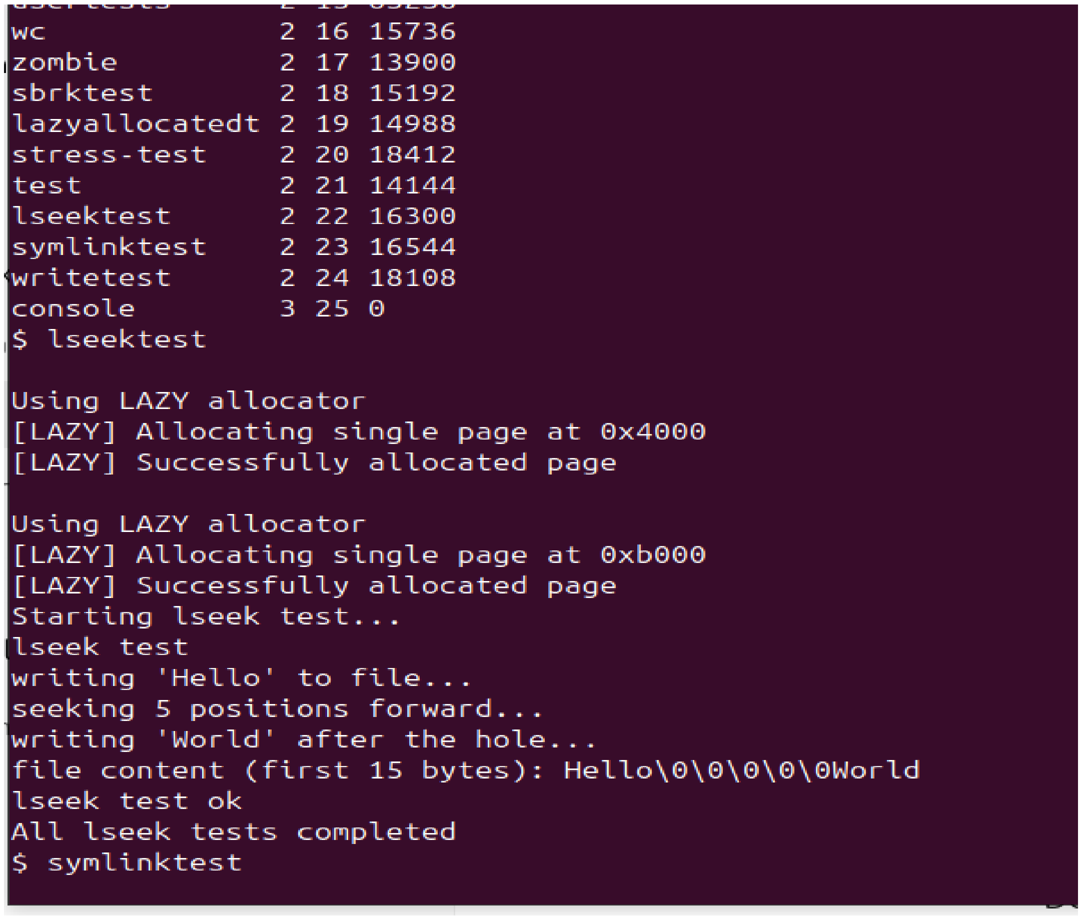
To implement the lseek() system call, we added a new system call entry in the syscall table and created the corresponding function sys_lseek() in sysfile.c.

Testing:

Make clean

Make qemu

lseektest



```
wc             2 16 15736
zombie         2 17 13900
sbrktest       2 18 15192
lazyallocatedt 2 19 14988
stress-test    2 20 18412
test           2 21 14144
lseektest      2 22 16300
symlinktest    2 23 16544
writetest      2 24 18108
console        3 25 0
$ lseektest

Using LAZY allocator
[LAZY] Allocating single page at 0x4000
[LAZY] Successfully allocated page

Using LAZY allocator
[LAZY] Allocating single page at 0xb000
[LAZY] Successfully allocated page
Starting lseek test...
lseek test
writing 'Hello' to file...
seeking 5 positions forward...
writing 'World' after the hole...
file content (first 15 bytes): Hello\0\0\0\0\0World
lseek test ok
All lseek tests completed
$ symlinktest
```

2. Adding Support for Symbolic Links

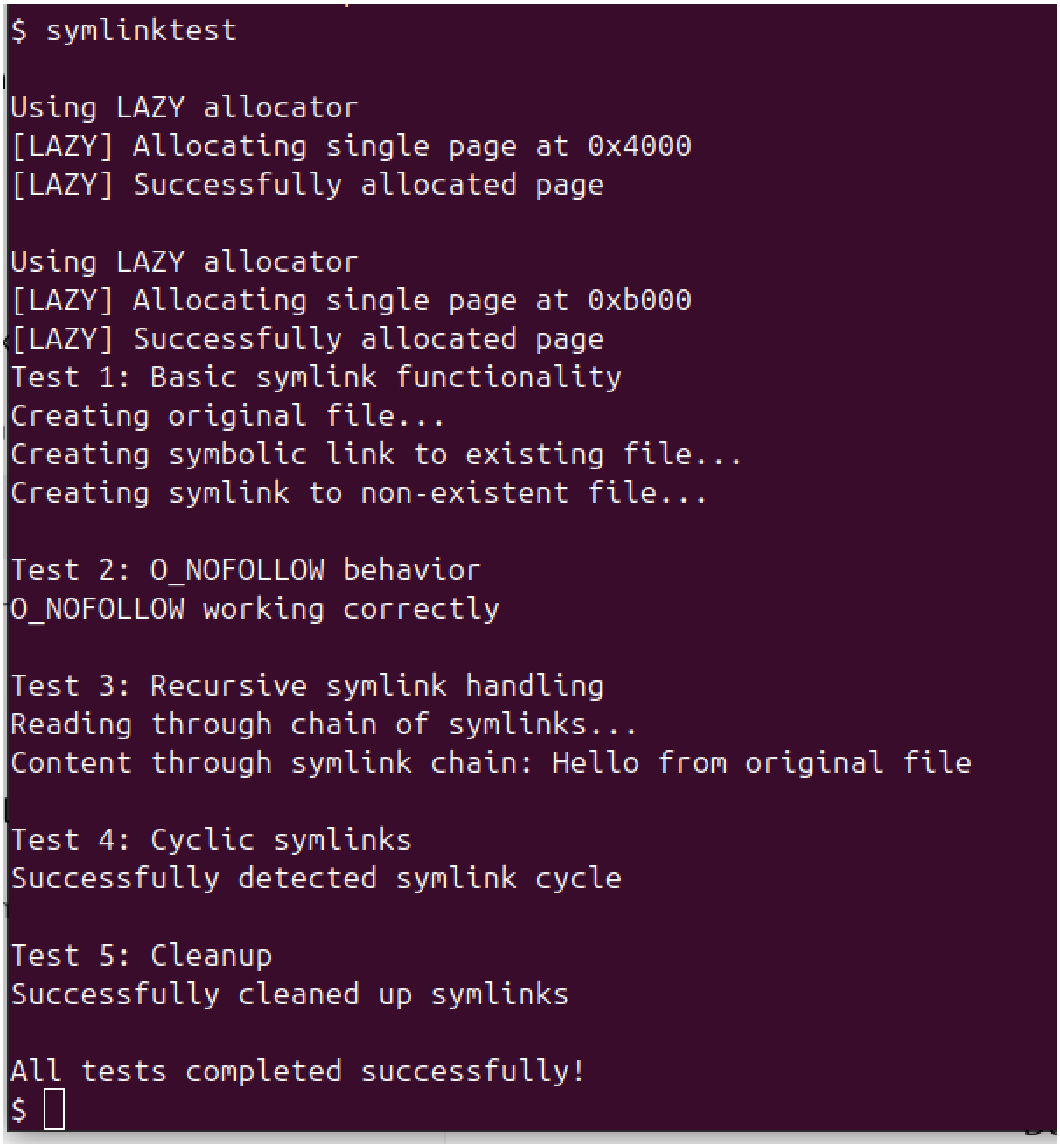
We implemented symbolic links by introducing a new file type T_SYMLINK and storing the target path within the inode's data blocks.

Testing:

Make clean

Make qemu

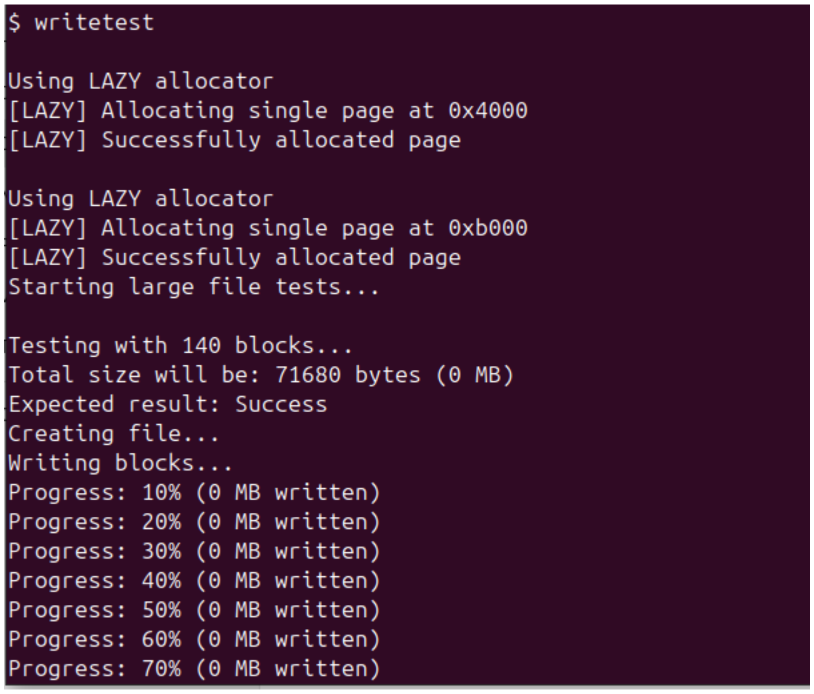
symlinktest



```
$ symlinktest

Using LAZY allocator
[LAZY] Allocating single page at 0x4000
[LAZY] Successfully allocated page

Using LAZY allocator
[LAZY] Allocating single page at 0xb000
[LAZY] Successfully allocated page
Test 1: Basic symlink functionality
Creating original file...
Creating symbolic link to existing file...
Creating symlink to non-existent file...

Test 2: O_NOFOLLOW behavior
O_NOFOLLOW working correctly

Test 3: Recursive symlink handling
Reading through chain of symlinks...
Content through symlink chain: Hello from original file

Test 4: Cyclic symlinks
Successfully detected symlink cycle

Test 5: Cleanup
Successfully cleaned up symlinks

All tests completed successfully!
$
```

3. Adding support for large files

We modified the file system to support two double-indirect blocks in each inode, increasing the maximum file size.



```
$ writetest

Using LAZY allocator
[LAZY] Allocating single page at 0x4000
[LAZY] Successfully allocated page

Using LAZY allocator
[LAZY] Allocating single page at 0xb000
[LAZY] Successfully allocated page
Starting large file tests...

Testing with 140 blocks...
Total size will be: 71680 bytes (0 MB)
Expected result: Success
Creating file...
Writing blocks...
Progress: 10% (0 MB written)
Progress: 20% (0 MB written)
Progress: 30% (0 MB written)
Progress: 40% (0 MB written)
Progress: 50% (0 MB written)
Progress: 60% (0 MB written)
Progress: 70% (0 MB written)
```