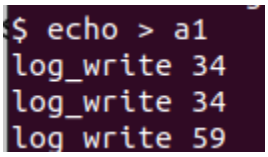
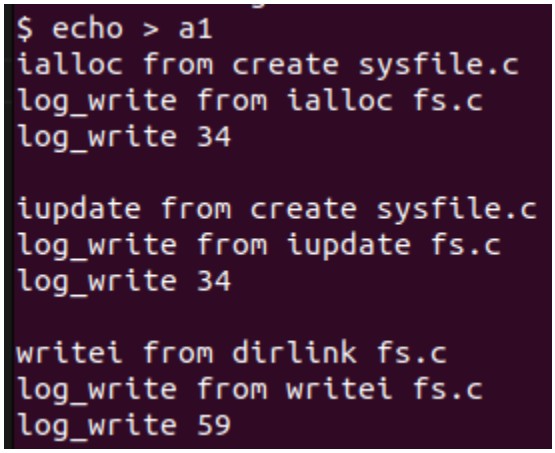
HOMEWORK: FILE-SYSTEMS
CO21BTECH11004

Part-1:

$ echo > a1



After adding some cprintf statements



When the above command is executed,

- create function is called from sysfile.c, which calls ialloc function which finds a free inode and allocates it for file a1 and returns struct pointer to that inode. It then writes log to block number 34.
- In the same create function iupdate is called to update (block 34), which updates the modified inode to the disk. Here modifications are done to ip->nlink=1, ip->major, and ip->minor are set after create function calls ialloc and before create function calls iupdate. Log_write 34.
- Dirlink function is called to write a new directory entry, which calls writei function that writes an empty data block for the new file. Log_write 59.

$ echo x > a1

```
$ echo x > a1
log_write 58
log_write 644
log_write 644
log_write 34
log_write 644
log_write 34
```

After adding some cprintf statements

```
$ echo x > a1
balloc from bmap fs.c
log_write from balloc fs.c
log_write 58

log_write from bzero fs.c
log_write 644

log_write from writei fs.c
log_write 644

iupdate from writei fs.c
log_write from iupdate fs.c
log_write 34

log_write from writei fs.c
log_write 644

iupdate from writei fs.c
log_write from iupdate fs.c
log_write 34
```

- Filewrite function is called, which calls writei (with off=0) which calls that calls bmap function which calls balloc function to allocate a new data block (644) for writing data(“x”), and calls bzero to zero out the data block(644) before writing. Now data is written to that data block(644), and iupdate is called, which updates the block's data.
- Filewrite function is called again, which calls writei (with off=1), do write and calls iupdate function.

$ echo xxx > a1

```
$ echo xxx > a1
log_write 644
log_write 644
log_write 644
log_write 34
log_write 644
log_write 34
```

After writing cprintf statements

```
$ echo xxx > a1
log_write from writei fs.c
log_write 644

log_write from writei fs.c
log_write 644

log_write from writei fs.c
log_write 644

iupdate from writei fs.c
log_write from iupdate fs.c
log_write 34

log_write from writei fs.c
log_write 644

iupdate from writei fs.c
log_write from iupdate fs.c
log_write 34
```

- Filewrite calls write a function with offset = 0,1,2 to write to block number 644 three times, now iupdate is called to update the block data(“xxx”).
- Here balloc, bzero are not called because we have already allocated a block (644) in the previous command to write “x”.
- Filewrite calls the write function with offset=4, to write the above data in the file, and update calls iupdate function to update data the same in the inode.

$ rm a1

```
$ rm a1
log_write 59
log_write 34
log_write 58
log_write 34
log_write 34
```

After adding cprintf statements

```
$ rm a1
writei from sys_unlink sysfile.c
log_write from writei fs.c
log_write 59

iupdate from sys_unlink sysfile.c
log_write from iupdate fs.c
log_write 34

bfree from itrunc fs.c
log_write from bfree fs.c
log_write 58

iupdate from itrunc fs.c
log_write from iupdate fs.c
log_write 34

iupdate from iput fs.c
log_write from iupdate fs.c
log_write 34
```

- Sys_unlink is called, which unlinks the blocks creates to store data from bmap by writing to bmap and calling iupdate to update the same.
- Itrunc function is called to remove the data associated with the file. It deallocates the datablock/s using bfree (644,645,646) function and updatei is called to update the same.
- iput is called to release the innode of the file, it decreases the ip->nlink by one and makes it 0, which was incremented on the creation of the file, then iupdate is called to update the same.

$ echo y > a2

```
$ echo y > a2
log_write 34
log_write 34
log_write 59
log_write 58
log_write 644
log_write 644
log_write 34
log_write 644
log_write 34
```

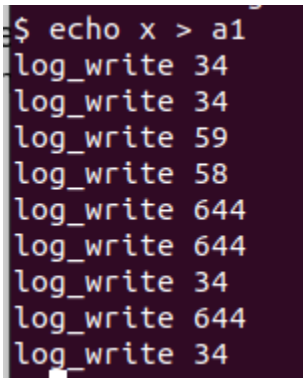
After adding cprintf statements

```
$ echo y > a2
ialloc from create sysfile.c
log_write from ialloc fs.c
log_write 34

iupdate from create sysfile.c
log_write from iupdate fs.c
log_write 34

writei from dirlink fs.c
log_write from writei fs.c
log_write 59

balloc from bmap fs.c
log_write from balloc fs.c
log_write 58

log_write from bzero fs.c
log_write 644

log_write from writei fs.c
log_write 644

iupdate from writei fs.c
log_write from iupdate fs.c
log_write 34

log_write from writei fs.c
log_write 644

iupdate from writei fs.c
log_write from iupdate fs.c
log_write 34
```

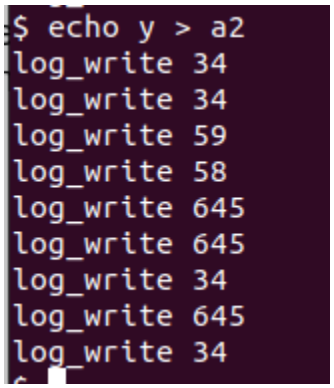
- Here first file a2 is created, same as part with the command “echo > a1” ,: - first three log_write: - (explained before)
    - Log_write 34: - ialloc
    - Log_write 34: - iupdate
    - Log_write 59: - writei
- Then “y” is written into file, same as part with command “echo x > a1”,: - next six log_write: - (explained before)
    - log_write 58: - balloc
    - Log_write 644: - bzero
    - Log_write 644: - writei
    - Log_write 34: - updatei
    - Log_write 644: - writei
    - Log_write 34: - updatei

Part-2

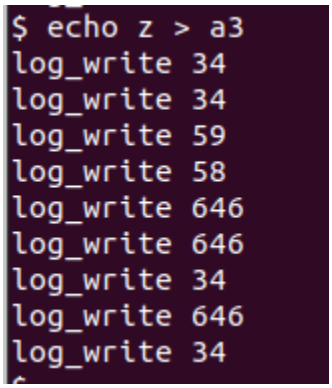
$ echo x > a1



$ echo y > a2

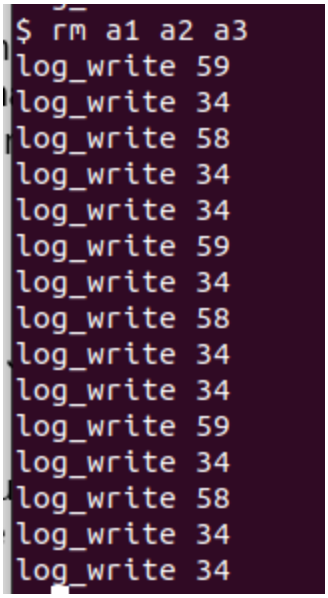


$ echo z > a3



- In all the above output, first three log_write, create function is called from sysfile.c, which calls ialloc function which finds a free inode and allocates it for the file and returns struct pointer to that inode.
- In all the above output block 34 is written and updated while creating the file,
- Then bmap entry is updated for the new block that is store the data for the file. It is same log_write 58 for writing bmap.
- For writing in different files, a new block is allocated with different block numbers (644, 645, 646), then zeroed using bzero and writei and iupdate to write and update.
- Then file is written and updated.

$ rm a1 a2 a3



- Here we can see repetition of output for “rm a1” explained before three times.
- Similarity,
    - for unlinking the file data by writing and updating, (log_write 59, log_write 34);
    - for writing updating bmap, log_write 58, log_write 34;
    - for updating file (iput) , log_write 34;

- Similarity is there as in same parent block inodes are searched and allocated for files.
- Each time Sys_unlink is called, which unlinks the blocks created to store data from bmap by writing to bmap and calling iupdate to update the same for the current file. (log_write 59, log_write 34).
- Then itrunc function to bfree to free block 644,645,646 and iupdate to remove data associated with the file and update it.
  (log_write 58, log_write 34)
- Then iput is called to release the innode of the file, it decreases the ip->nlink by one and makes it 0, which was incremented on the creation of the file, then iupdate is called to update the same.
  (log_write 34)