ECE 391 Exam 2, Fall 2014

Monday, November 10, 2014

Name:____

NetID:____

Discussion Section:

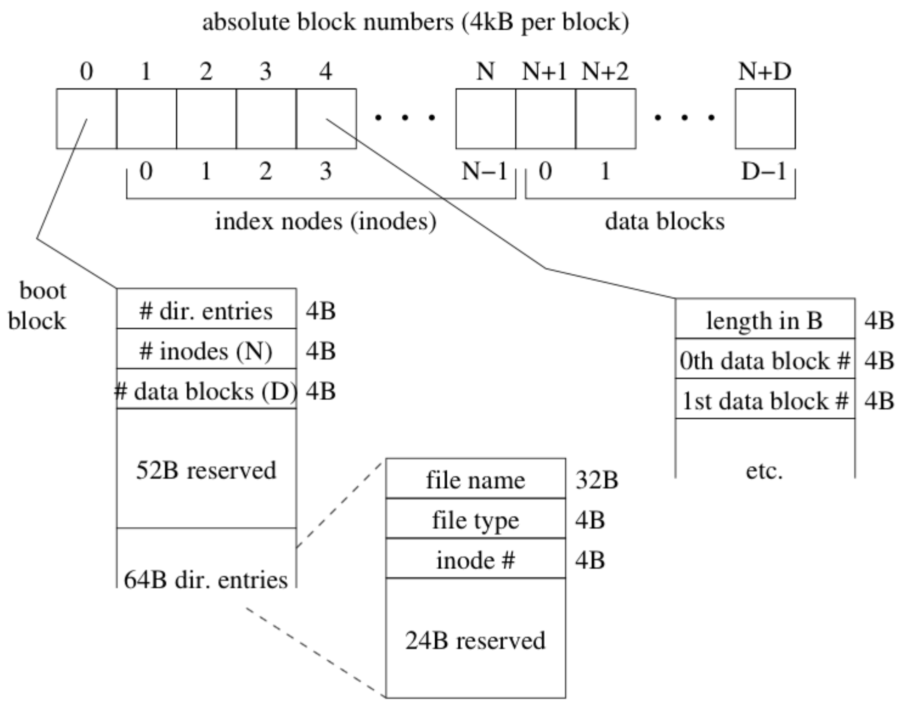
- ◯ 10 a.m. (Eric Badger)
- ◯ 11 a.m. (Matt Tischer)
- ◯ 1 p.m. (Matt Tischer)
- ◯ 2 p.m. (Yixiao Nie)
- ◯ 3 p.m. (Ying Chen)

- Be sure that your exam booklet has 10 pages.
- Write your net ID at the top of each page.
- This is a closed book exam.
- You are allowed two $8\,{}^1\!/_2 \times 11$" sheet of notes.
- Absolutely no interaction between students is allowed.
- Show all of your work.
- Don't panic, and good luck!

Page	Points	Score
3	9	
4	7	
5	10	
6	4	
7	5	
8	8	
9	5	
10	14	
Total:	62	

Question 1: File System Fairy Tale . . . . 16 points

In a land far away a file system stood. One from MP3, something to expect, you should. It was a magical place with wonder infused, yet there was one man who was a tad bit confused. He was taking a class, the workload a ton. This class, indeed, was 391. The file system had inodes, files, and even a byte, which the brilliant student just could not get right. He struggled for awhile, he was not a fan. The confusion continued until he developed a plan. He remembered his family, forever matriarchal, whose mother just happened to be the pony Twilight Sparkle. She had some connections to some students who knew, everything that the student would have to do. The student was pleased and thought it was great, anxiously hoping it wasn't too late. The information finally landed him here, where he requires you students to displace all his fear. If you finish his quest and finish it well, you will be rewarded with points and all will be swell. So grab your quill, pencil, or pen, and finish every task, requested by Ben.



Notes:

1KB = 1024 B

You may leave expressions in unsimplified form.

Assume only regular files and directories (e.g. no devices).

(a) (2 points) What is the maximum number of files that this file system can support?

(b) (2 points) What is the maximum size of any single file in this file system?

(c) (5 points) Assume that your file system contains the maximum number of files and that each of those files is 4B. What fraction of the total filesystem size is used for actual data?

(d) (2 points) Why do the reserved bytes exist in the boot block as well as in each of the directory entries?

(e) (5 points) Describe the process of reading a file from this file system step-by-step

(f) (5 points (bonus)) Implementing write on this filesystem would be inefficient. Explain why and suggest an additional file system data structure that would help.

Question 2: Paging Tradeoffs . . . . 14 points

(a) (10 points) Ben Bitdiddle wants to explore the tradeoffs between different (hypothetical) paging schemes. Help him fill in the tables below. One row is filled in for you as an example. Assume that the index bits are divided equally between all levels and that the size of one entry in a paging structure is always 4 bytes.

In the last row, you may assume that s is a power of 2 and that n and s are chosen such that the number of index bits (per level) is an integer.

# Levels	Size of Pages	# Page Offset Bits	# Index Bits (/level)
2	4kB	12	10
20	4kB		
1	1GB		
2	4B		
n	s		

# Levels	Size of Pages	Size of One Entry	# Entries (/level)	Size of Paging Data (/level)
2	4kB	4B	1024	4kB
20	4kB	4B		
1	1GB	4B		
2	4B	4B		
n	s	4B		

(b) (2 points) Why would we choose the paging scheme with one level of 1GB pages over the scheme with 20 levels of 4kB pages?

(c) (2 points) Why would we choose the paging scheme with two levels of 4B pages over the scheme with one level of 1GB pages?

Question 3: TLB(caching) .... 18 points

Consider a virtual memory system with 4 KB pages, and a TLB that has 3 translation entries. The TLB uses a "least recently used" (LRU) replacement policy. A program operates on a 2-dimensional array of 4×1024 32-bit integers, `int matrix[4][1024]`. The array is stored consecutively in memory, starting at virtual address 0xc80000. Table 1 shows the layout of the array.

Virtual address	Data
0xc80000	matrix[0][0]
0xc80004	matrix[0][1]
0xc80008	matrix[0][2]
...	...
0xc81000	matrix[1][0]
0xc81004	matrix[1][1]
...	...

Table 1: matrix array storage in memory

Consider the following code:

```
int x, y;
result = 0;
for (x = 0; x < 1024; x++) {
  for (y = 0; y < 4; y++) {
    result += matrix[0][x]*matrix[y][x];
  }
}
```

(a) (5 points) Show the state of the TLB after each iteration of the inner loop, i.e., after the `result += ...` instruction has been executed. For each TLB translation entry, write down the virtual address of the corresponding page. Assume that the TLB starts out empty, and that the variables x, y, and result are stored in registers and thus do not require memory accesses; likewise, instruction fetches do not use the TLB. The table captures the first six iterations, with the first column already filled in for you.

	round 1 (x=0, y=0)	round 2 (x=0, y=1)	round 3 (x=0, y=2)	round 4 (x=0, y=3)	round 5 (x=1, y=0)	round 6 (x=1, y=1)
TLB entry 1	0xc80000					
TLB entry 2	empty					
TLB entry 3	empty					

(b) (3 points) What is the number of TLB misses that will occur during the execution of the entire program (i.e., 4×1024 rounds).?

(c) (5 points) Rewrite this code to perform the same calculation with fewer TLB misses

(d) (2 points) How many TLB misses will occur during the execution of your revised code?

(e) (3 points) Your partner, Ben Bitdiddle, has configured the scheduler so that there is a context switch after every inner loop iteration. He notes, with pride, that your optimized code and the original now both incur the same number of TLB misses. Explain why.

Question 4: Scheduling . . . . 14 points

(a) (4 points) Give one example each of a IO-bound and a compute-bound process.

(b) (4 points) Which of the two processes should have higher priority and why?

(c) (2 points) How long is a typical time slice (quantum)? Circle the correct choice.

A. 10–100 ns

B. 10–100 ms

C. 10–100 s

(d) (4 points) Describe one advantage and one disadvantage of making a quantum larger.