Department of Computer Science
Operating Systems Qualifying Exam
Fall, 2005

The following exam is open book and open notes. You may feel free to use whatever additional reference material you wish, but **no electronic aids** are allowed. Please note the following instructions. There will be a ten point deduction for failure to comply with them:

- show your work whenever appropriate. There can be no partial credit unless we are able to see how you arrived at your answers.
- be succinct. You may lose points for facts that, while true, are not relevant to the question at hand

Questions 3 and 4 include tables, which you may use for your answer if you so choose. If so, please make sure you include the table with your blue book!

1. (30 points) Suppose you have a computer system with a 64 K byte, 2-way set-associative data cache with a 16 byte line using LRU replacement using physical addresses, and you are executing the following code:

```
for (i = 0; i < 1024; i++)
    a[i] = b[i] + c[i];
```

   Array `a` starts at virtual address `0x00010000`, `b` starts at `0x00020000`, and `c` starts at `0x00030000`. `a`, `b`, and `c` are all arrays of 4-byte integers. The index variable `i` is kept in a register.

   Assume the computer's virtual memory system uses a 4K page.

   In the following, consider only cache behavior: assume there are no page faults.

   (a) Explain how, if the operating system chooses an unlucky mapping from virtual to physical space, the data cache hit rate can be approximately 0% (no hits whatever).

   (b) Explain how the operating system can set up the mapping from virtual to physical space to make the data cache hit rate as high as possible, and estimate the hit rate.

2. (30 points) Suppose you have a Unix-like file system whose i-nodes contain 12 direct pointers, an indirect pointer, and a double indirect pointer. If a block is 4K bytes and a file block pointer is 32 bits,

   (a) What is the maximum size of the file system?

   (b) What is the maximum size of a file?

   (c) How many disk accesses are required to read byte number $\texttt{123abc789}_{16}$ of a file, starting with a read of the file's i-node (disregard any caching of i-nodes and blocks)?

3. (30 points) For the following problem, assume an Alewife (*ie* LimitLESS protocol) computer system. Assume variable `x` is homed (has its directory entries) on process P_1 and has a processor set consisting of $\{P_2, P_3\}$, and `y` is homed on P_2, is in read-write state, and has a processor state consisting of $\{P_1\}$. Now, suppose processor P_3 executes the following line of code:

```
x = y;
```

Tell all the messages that need to be transferred between P_1, P_2, and P_3 to accomplish this. For each message, tell the source and destination processors, the message type, the variable (`x` or `y`), and the transition (as given in Table 3 of the Alewife paper) that occurs as a result.

Message Type	Variable	Source Processor	Destination Processor	Transition Number	Comments

4. (10 points) Some computer system is running a mix of hard real-time, multimedia, and batch processes.

The real-time processes have the following arrival times, processing requirements, and deadlines:

Process	Arrival Time	Processing Time	Deadline
R_1	0	5	10
R_2	5	7	15

The multimedia processes have the following processing requirements:

Process	Processing Time per Epoch	Epoch Length
M_1	1	5
M_2	2	10

Finally, the two batch processes are B_1 and B_2.

Schedule these processes to meet the following constraints:

- Both of the hard real-time processes must meet their deadlines.
- The multimedia processes should come as close to receiving their time allotments as possible subject to the first constraint. If a process can not be given its total allotment in an epoch, it should not be given any time in that epoch.
- The two batch processes should be given any left-over cycles in a round-robin allocation.

Process	Cycle																			
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
B_1																				
B_2																				
M_1																				
M_2																				
R_1																				
R_2																				