Please enter your name and Athena login name in the spaces above. Enter your answers in the spaces provided after each question. You can use the extra white space and the backs of the pages for scratch work.

**Problem 1. Potpourri (5 points)**

(A) Semaphore S is used to implement mutual exclusion on accesses to a shared buffer. No other semaphores are used. What should its initial value be?

Initial value for S: __________

(B) You are using a semaphore to ensure that no more than 3 of your company’s 16 printers are in use at the same time. What should its initial value be?

Initial value of semaphore: __________

(C) A Beta processor has an interrupt handler invoked by a periodic 60Hz clock. The kernel-mode handler simply inspects the high-order bit of the XP register to see if it is a 1 or 0. Give your best estimate of the fraction of the time it finds a 1.

Circle one: 0% ... 50% ... 100%

(D) If a user-mode application uses the XP register, this may prevent interrupts from returning to the proper location.

Circle one: TRUE ... FALSE

(E) An MMU incorporates a page map with $2^{24}$ entries, where each entry (including D and R bits) has 18 bits. If the page size is cut in half without changing the size of the virtual address or the physical address, what are the new dimensions of the page map?

Number of page map entries: __________, number of bits in each entry: __________
Problem 2. Virtual Memory (7 points)

Consider an MMU that supports a page size of $2^{12}$ bytes. The page map has $2^{18}$ entries and each page entry requires 32 bits, including the D and R bits.

(A) (2 Points) Please give the number of bits in the virtual address that is sent to the MMU and in the physical address produced by the MMU. Write “Can’t tell” if the size of an address cannot be determined.

| Number of bits in the virtual address: __________ |
| Number of bits in the physical address: __________ |

A snapshot of the first 8 locations of the page map is shown to the right. Note that the least recently used page is virtual page 6 and the next least recently used page is virtual page 4. Then the MMU is used to perform the address translations required when executing the following instruction which is located at physical address 0x1234:

\[
\text{ST(R31, 0x1234, R31)}
\]

(B) (1 Point) What is the virtual address of the ST instruction?

Virtual address of ST instruction: 0x_______

(C) (4 Points) The ST instruction writes to virtual address 0x1234. Please make the appropriate changes to the page table above to show its contents after the execution of the ST instruction. Immediately after execution of the ST, what is the physical address of the memory location that was written by the ST instruction, i.e., the physical address of the location whose virtual address is 0x1234?

Modify contents of the page table above

Physical address written by the ST instruction: 0x_______
Problem 3. Real Time (5 Points)

A real-time operating system is using a *strong priority system* to manage the execution of three tasks – A, B, and C – whose specifications are given in the table below. *Service time* indicates the time required to run each kernel-mode task handler. *Deadline* indicates the maximum time that can elapse between the request and the completion of the task handler. *Period* indicates the interval between the recurring requests for each task.

<table>
<thead>
<tr>
<th>Task</th>
<th>Service time</th>
<th>Deadline</th>
<th>Period</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>20us</td>
<td>25us</td>
<td>50us</td>
</tr>
<tr>
<td>B</td>
<td>100us</td>
<td>500us</td>
<td>1000us</td>
</tr>
<tr>
<td>C</td>
<td>5us</td>
<td>15us</td>
<td>25us</td>
</tr>
</tbody>
</table>

Using the earliest-deadline scheduling algorithm, the system engineers have determined that the appropriate priority order is C > A > B, i.e., C has the highest priority and B the lowest.

(A) (2 Points) The system is also used to execute user-mode applications when not servicing task requests. If a user-mode program takes 10 seconds to complete when task requests are disabled, how long will the program take to complete when task requests are enabled and occur with the periods and service times given above?

*Time to execute P when task requests are enabled (seconds): *

(B) (3 Points) Assume that tasks are enabled and requests are occurring at intervals specified by the “period” column in the table. What is the worst-case completion time for each of the task handlers?

*Worst-case completion time for task A (us): *

*Worst-case completion time for task B (us): *

*Worst-case completion time for task C (us): *
Problem 4. Concurrency & Synchronization (8 Points)

Processes A and B, whose code is shown on the right, share a common variable X. X is set to 5 before either process begins execution. As usual, statements within a process are executed sequentially, but statements in process A may execute in any order with respect to statements in process B.

After all the statements in both processes have executed, there are four possible ending values for X. Here are the possible ways in which statements from A and B can be interleaved:

- A1 A2 B1 B2: X = 11
- A1 B1 A2 B2: X = 6
- A1 B1 A2 B2: X = 10
- B1 A1 A2 B2: X = 6
- A1 B1 B2 A2: X = 10
- B1 B2 A1 A2: X = 12

In the questions below, semaphores have been added to the code. For each modified program, please indicate all the possible ending values of X after all the statements that will be executed have been executed.

(A) Ending value(s) for X: ____________

(B) Ending value(s) for X: ____________
(C) Ending value(s) for X: __________

```
shared int X = 5;
semaphore S = 1;

Process A            Process B
int Y;                int Z;
wait(S);             wait(S);
A1: Y = X*2;         B1:  Z = X+1;
A2: X = Y;           B2:  X = Z;
signal(S);           signal(S);
```

(D) Ending value(s) for X: __________

```
shared int X = 5;
semaphore S = 0;
semaphore T = 0;

Process A            Process B
int Y;                int Z;
A1: Y = X*2;         B1:  Z = X+1;
signal(T);           signal(S);
wait(S);             wait(T);
A2: X = Y;           B2:  X = Z;
```
Problem 5. Cache Coherence (5 points)

Consider a multicore system where processors P1 and P2 each have a private, snooping-based, write-back cache that uses the MESI coherence protocol. Suppose the processors make a sequence of accesses to the shared variable A in the order listed below:

P1: LD A
P1: ST A
P2: LD A
P1: ST A
P2: ST A
P2: ST A

Please fill in the table below showing the shared-bus operations (one of BusRd, BusRdX, or BusWB) required to complete each access. If no shared-bus transaction is required for the access, leave the entry blank. Also give the state of the cache line for location A in each of P1’s and P2’s cache after the access is complete (one of M, E, S, or I)

<table>
<thead>
<tr>
<th>Access</th>
<th>Shared bus transaction(s)</th>
<th>P1’s cache state for A</th>
<th>P2’s cache state for A</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial state</td>
<td></td>
<td>I</td>
<td>I</td>
</tr>
<tr>
<td>After P1: LD A</td>
<td>P1: BusRd(A)</td>
<td>E</td>
<td>I</td>
</tr>
<tr>
<td>After P1: ST A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>After P2: LD A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>After P1: ST A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>After P2: ST A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>After P2: ST A</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Here’s the state transition diagram for the MESI cache coherence protocol:

| END OF QUIZ 4! Have a good summer… |