University of Waterloo
Midterm Examination
Term: Fall Year: 2007

Student Name	______________________
Student ID Number	______________________

Course Abberviation and Number:	CS 350
Course Title:	Operating Systems
Section(s):	1
Instructors:	Tim Brecht

Date of Exam:	October 30, 2007
Time Period	Start time: 1:00 pm End time: 2:20 pm
Duration of Exam:	80 minutes
Number of Exam Pages:	9 (including cover sheet)
	NO CALCULATORS, NO ADDITIONAL MATERIAL

Marking Scheme:

Problem	Mark	Score
1	13	
2	8	
3	14	
4	10	
5	9	
6	6	
Total	60	

Problem 1 (13 marks)

a. **(4 mark(s))** Explain two different methods that could be used to supply synchronization primitives to user-level threads to allow them to synchronize with each other and discuss the pros and cons of the two approaches.

b. **(5 mark(s))** Mark each of the following as either: (I) interrupt, (E) exception, or (N) neither.

(a) Timer

(b) Procedure Call

(c) Segmentation Violation

(d) Divide by Zero

(e) System Call

c. **(2 mark(s))** What is the difference between deadlock and starvation?

d. **(2 mark(s))** A multi-threaded program running on a system with a *single processor* that does not synchronize access to shared variables will result in data corruption or some other incorrect behavior?

- (a) Always
- (b) Sometimes
- (c) Never

Explain your answer:

Problem 2 (8 marks)

a. **(2 mark(s))** Under what conditions might two or more processes share common physical frames. Explain how and why this would be done?

b. **(2 mark(s))** Name one of the key operating system abstractions that user-level threads from the same process must share and explain why it is shared.

c. **(2 mark(s))** Explain what is meant by the context of a thread?

d. **(2 mark(s))** In OS/161 different parts of a thread's context can be saved and restored at different points during the kernel's execution. Describe where these different points occur (logically) and why they are necessary.

Problem 3 (14 marks)

There have been some accidents recently on the one-lane bridge near Conestogo. To prevent future accidents, traffic lights have been installed at either end of the bridge to synchronize the traffic going in different directions. A car can only cross the bridge if there are no cars going the opposite direction on the bridge. Sensors at either end of the bridge detect when cars arrive and depart from the bridge, and these sensors control the traffic lights. Below is a skeleton implementation of two routines, Arrive() and Depart(). You may assume that each car is represented by a thread, and threads call Arrive() when they arrive at the bridge and Depart() when the leave the bridge. Threads pass their direction of travel as input to the routines.

```
#define DIR_OPEN   (0)
#define DIR_NORTH  (1)
#define DIR_SOUTH  (2)
```

```
struct lock *lock = lock_create("Lock");
struct cv *cv = cv_create("CV");
int curdir = DIR_OPEN; int numcars = 0;
```

```
    Arrive(int mydir)
    {
A.1
A.2   while (curdir != mydir &&
A.3          curdir != DIR_OPEN)  {
A.4
A.5      ; /* spin doing nothing */
A.6
A.7    }
A.8
A.9    numcars++;
A.10
A.11   curdir = mydir;
A.12
     }
```

```
    Depart(int mydir)
    {
D.1
D.2     numcars--;
D.3
D.4     if (numcars == 0) {
D.5
D.6        dir = DIR_OPEN;
D.7
D.8     }
D.9
D.10
D.11
D.12
     }
```

a. **(2 mark(s))** The code above doesnt't properly synchronize access to shared resources. Outline an execution sequence (referring to the numbered statements) where two threads can cause two cars to travel on the bridge in opposite directions at the same time.

b. **(6 mark(s))** Show how the declared condition variable and lock could be used to correctly synchronize the cars by annotating and/or modifying the above code with calls to Condition Variable and Lock operations.

c. **(2 mark(s))** In your solution, draw arrows connecting the start and end of all the critical sections of code.

continued on the next page ...

d. **(2 mark(s))** Can your solution lead to starvation? Briefly explain.

e. **(2 mark(s))** Can your solution lead to deadlock? Briefly explain.

The space below is intentionally left blank.

Problem 4 (10 marks)
CHOOSE AND ANSWER ONLY TWO OF THE THREE PARTS OF THIS QUESTION.
IF YOU ANSWER ALL THREE PARTS, THE LOWEST TWO SCORES WILL BE USED.
Some useful info: $2^{10} = 1$ KB, $2^{20} = 1$ MB, $2^{30} = 1$ GB

a. **(5 mark(s))** In this part of the question all addresses, virtual page numbers and physical frame numbers are represented in hexidecimal, recall that each hexidecimal character represents 4 bits. Consider a machine with *48-bit* virtual addresses and a page size of 16 MB. During a program execution the TLB contains the following valid entries (in hexidecimal).

Virtual Page Num	Physical Frame Num
0x 0	0x 1
0x BB	0x 2
0x BB9	0x 3
0x BB97	0x 4
0x BB972	0x 5
0x BB9720	0x 6

Translate the following virtual address into a *52-bit* physical address (in hexidecimal). Show and explain how you derived your answer. Express the final physical address using all 52-bits. Virtual address = 0x 00BB 9720 AF05.

b. **(5 mark(s))** In this part of the question all addresses, virtual page numbers and physical frame numbers are represented in octal, recall that each octal character represents 3 bits. Consider a machine with *24-bit* virtual addresses and a page size of 512 bytes. During a program execution the TLB contains the following valid entries (all in octal).

Virtual Page Num	Physical Frame Num
0	10
7	20
73	30
731	40
7310	50

Translate the following virtual address (in octal) into a *24-bit* physical address (in octal). Show and explain how you derived your answer. Express the final physical address using all 24-bits. Virtual address = 07310525.

c. **(5 mark(s))** In this part of the question all addresses, virtual page numbers and physical frame numbers are represented in decimal. Consider a machine with *32-bit* virtual addresses and a page size of 512 bytes. During a program execution the TLB contains the following valid entries (all in decimal).

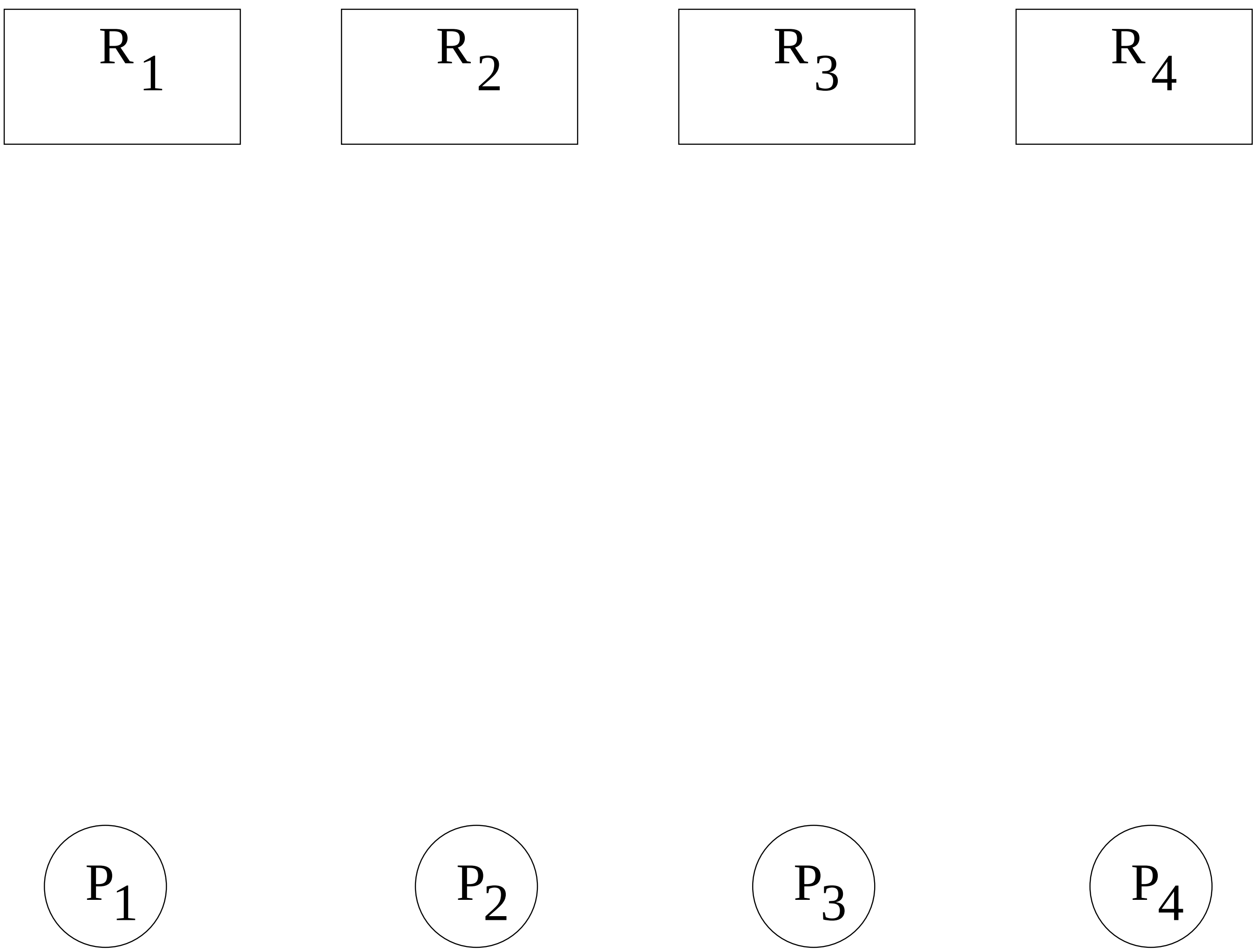
Virtual Page Num	Physical Frame Num
591	100
5912	200
11548	300
2589	400
59125	500

Translate the following virtual address (in decimal) into a physical address (in decimal). Show and explain how you derived your answer. Express the final physical address using decimal digits only. Virtual address = 5912589.

Problem 5 (9 marks)
This question uses the following notation (as used in the course notes) to describe resource allocation in a computer system :

- R_i: request vector for process P_i
- A_i: current allocation vector for process P_i
- U: unallocated (available) resource vector

Given the scenarios below, draw the corresponding resource allocation graph. PLEASE USE SOLID LINES WITH ARROWS FOR ALLOCATION EDGES AND DASHED OR DOTTED LINES WITH ARROWS FOR REQUEST EDGES. Indicate if the system is deadlocked and justify your answer (one sentence).
$U = (0, 1, 0, 0)$
$R_1 = (1, 0, 0, 0)$, $R_2 = (0, 0, 1, 0)$, $R_3 = (0, 1, 0, 0)$, $R_4 = (0, 0, 0, 1)$
$A_1 = (1, 0, 0, 1)$, $A_2 = (1, 1, 0, 0)$, $A_3 = (1, 0, 0, 0)$, $A_4 = (0, 1, 1, 1)$.



Problem 6 (6 marks)

a. **(3 mark(s))** Write a small C code fragment that when run on OS/161 would generate an "Address Error on Load" exception (USING AN ADDRESS THAT IS NOT PART OF THE KERNEL'S ADDRESS SPACE). Comment your code to explain what it is doing.

b. **(3 mark(s))** Write a small mips assembly code fragment that makes an OS/161 system call with the value 57 as an argument to the system call numbered 7. You are not required to worry about the return value of the system call, to have correct syntax, or recall all instruction names precisely; pseudo-assembly language is fine. Comment your code to explain what it is doing.