Roll Num: Name:

CS3523: Operating Systems 2

Quiz2 – Spring 2022

Process Synchronization

1. In a system there exists multiple threads of two types:
   a) hydrogen b) oxygen

   A. Each hydrogen thread waits until it's made into a water molecule to complete execution. In other words, it waits for another hydrogen and one oxygen molecule
   B. Each oxygen thread waits for two other hydrogens to make a water molecule.

Design procedures for Hydrogen and Oxygen threads that use monitors for synchronization **[6 pts]**

2. Some platforms provide a pair of instructions that work in concert to help build critical sections. On the MIPS architecture, for example, the load-linked and store-conditional instructions can be used in tandem to build locks and other concurrent structures. The C pseudocode for these instructions is shown below. Alpha, PowerPC, and ARM provide similar instructions.

```
// Actions of thread Ti
1: int LoadLinked(int *ptr) {
2:      return *ptr;
3: }

4: int StoreConditional(int *ptr, int value) {
5:      if (no other thread has updated *ptr since the last LoadLinked access to *ptr by thread Ti)
6:      {
7:              *ptr = value;
8:              return 1; // success!
9:      }
10:     else {
11:             return 0; // failed to update
12:     }
13: }
```

The LoadLinked instruction operates much like a typical load instruction. It simply returns the current value of the memory location.

The key difference comes with the StoreConditional. It usually works in tandem with LoadLinked instruction. The normal sequence of invoking StoreConditional instruction is to first invoke LoadLinked which is then followed by StoreConditional instruction.

StoreConditional only succeeds (and updates the value stored at the address just load-linked from) if no intervening StoreConditional by another thread to the same address takes place. In the case of success, the store-conditional returns 1 and updates the value at ptr to value; if it fails, the value at ptr is not updated and 0 is returned.

If StoreConditional is invoked by itself without invoking the corresponding LoadLinked instruction, then the StoreConditional returns false.

Please develop a mutex lock using this instruction. **[5 pts]**

3. The book has shown the implementation of monitors - hold and wait using semaphores. Can you please develop a solution for implementing monitors for hold and continue case using semaphores. **[10 pts]**

4. C++ atomics provides an operation: *fetch_xor*. This instruction atomically applies bitwise XOR to the contained value, say *x*. Can you please provide an implementation of this operation using CAS operation.

Some clarification for this question - multiple threads are updating a shared value *x*. When this operation fetch_xor is applied on *x*, this instruction atomically applies a bitwise XOR on *x.* **[4 pts]**

5. We studied the implementation of Signal & Wait monitor implementation using semaphores. In this implementation, please explain in detail the role of the 'next' semaphore. The implementation is shown below. Specifically, please explain the lines: 6, 11, 19.
For each function f:

```
1: wait(mutex);
2: ...
3: body of F
4: ...
5: if (next_count > 0)
6: signal(next);
7: else
8: signal(mutex);

x.wait():

9: x count++;
10: if (next_count > 0)
11: signal(next);
12: else
13: signal(mutex);
14: wait(x_sem);
15: x count--;

x.signal():

16: if (x count > 0) {
17: next_count++;
18: signal(x_sem);
19: wait(next);
20: next_count--;
21:}
```

6. In the class, we discussed a solution to the dining philosopher's problem using monitors. But this solution is susceptible to process starvation. Please explain how?

[3 pts]

7. Placement interviews problem:
There is a representative from Google conducting campus interviews. If there are no students waiting for interviews he will be working on analyzing results. There are N chairs available outside the room for the students to wait if the representative is currently conducting an interview of another student. Design two processes, the representative and the student using semaphores such that:

A. Representative does other work if no students are present.
B. Student requests for an interview if the representative is doing other work.
C. Student waits in the chair if the representative is conducting an interview for others.
D. Student leaves if all the chairs are filled outside the room.

[8 pts]