Kevin Pham (kmp2996)

Jia-luen Yang (JY8435)

EE360P HW 1 Questions

**Question 3: Create a TACC UserID and submit it along with Homework questions.**

Kevin Pham: kpham95

Jia-luen Yang: jy8435

**Question 4: Show that any of the following modifications to Peterson’s algorithm makes it incorrect:**

**a) A process in Peterson’s algorithm sets the turn variable to itself instead of setting it to the other process.**

Assume there are two processes P0 and P1:

Initialization:

shared int turn = 0

bool wantCS[0] = false

bool wantCS[1] = false

P0 starts, wantCS[0] = true, and turn = 0. P0 enters critical section.

P1 interrupts and starts at this point. P1 sets wantCS[1] = true and turn = 1. P1 enters critical section.

Both processes enter critical section. This cannot happen because of mutual exclusion.

**b) A process sets the turn variable before setting the wantCS variable.**

Assume there are two processes P0 and P1:

P0 starts entry, turn = 1.

At this point, P1 also starts entry, wantCS[1] = true, turn = 0. Because wantCS[0] = true, P1 will enter critical section.

P0 then starts from the previous point, and wantCS[0] = true, and because turn = 0, P0 will also enter critical section.

Since both processes enter critical section, this is invalid.

**Question 5: Peterson’s algorithm uses a multi-write variable turn. Modify the algorithm to use two variables turn0 and turn1 instead of turn such that P0 does not write to turn1 and P1 does not write to turn0.**

Suppose turn0 and turn1 are in represented by array turn[]:

init:

bool turn[] = [false, false];

shared bool wantCS[] = [false, false];

requestCS:

wantCS[i] = true;

turn[i] = true;

while((wantCS[1 - i] && (turn[i] != turn[1 - i])) || (turn[i] == turn[1 - i] && (i < (1 - i))))

no-op();

CS

wantCS[i] = false;

turn[i] = false;

**Question 6: Show that the bakery algorithm does not work in the absence of choosing variables.**

Assume two processes P0 and P1:

P0 starts and goes through the first for loop.

P1 interrupts, sets number[1] to 1 during step one. Because number[1] is the biggest in the number array, P1 will enter critical section in step 2.

P0 then starts again, number[0] is set to 1 because it has already exited the first for loop in step 1. In step 2, number[0] == number[1] and 0 (process 0) < 1 (process 1), therefore P0 will also enter critical section.

At this point, P0 and P1 are both in critical section, which violates mutual exclusion, therefore the absence of the choosing variable would break bakery algorithm.