EE3609 HW1

Mikalai Shlapnikau
ms68887

3) Tacc UserID = mshlapno

4) a)

	P0	P1
1	want CS[0] = T	want CS[1] = T
2	turn = 0	turn = 1
3	while ((turn == 1) && want CS[1]) {}	while ((turn == 0) && want CS[0]) {}
4	CS	CS
5	want CS[0] = false	want CS[1] = F

If P0 executes 1, 2, then P1 executes 1, we have want CS[0] = T, want CS[1] = T, turn = 0. P0 then can enter CS. Then P1 goes to 2 and sets turn = 1, which allows it to enter CS as well. So we have 2 processes in CS at the same time, which is wrong.

b) If the turn variable is set before the want CS variable, the what we can get is this

1	turn = 1	turn = 0
2	want CS[0] = T	want CS[1] = T
3	while (turn == 1 & want CS[1])	while (turn == 0 & want CS[0])
4	CS	CS
5	want CS[0] = F	want CS[1] = F

If P0 executes 1, and P1 executes though the while loop and to the CS, P0 then can execute 2, 3 and fall into the CS as well. **2 threads in CS.**

5)

want CS[0] = true;	want CS[1] = true
local0 = turn1	local1 = 1 - turn0
turn0 = local0	turn1 = local1
while (want CS[1] && (local0 == turn1));	while (want CS[0] && (local1 != turn0))
CS	CS
want CS[0] = false	want CS[1] = false

6) Bakery algorithm would fail if it did not have choosing variables, consider P0 and P1:

P0 goes in the doorway, goes through the loop and finds that the max # is 0. However, it does not increment yet.

P1 then goes in and does the same thing, finding that the max # is 0, however it then continues through the second for loop, since the number is the only one with a value of 1 in the array, it enters CS.

P0 then continues, gets assigned number[0] = 1, and goes through the for loop, seeing that the number is the same as P1 but $1 \neq 0$ and so it goes through by having a smaller PID. P0 enters CS.

Now, both P0 and P1 are in CS, incorrect.