Samuel Villarreal

Sid: 861021551

Cs160 Concurrent Programming and Parallel Systems

CS 160: Lab Assignment 2

Due at 11:59PM on Feb 24, 2014

**1)**

File is included you can run test by calling: make test1

**2)**

File is included you can run test by calling: make test2

/\*

\* what was happening was that the main was exiting before the thread woke up

\* fix: waited for the thread created to finish before exiting main

\*/

**3)**

File is included you can run test by calling: make test3

Problem was that the main and new thread race to look at t

**4)**

A)

V(t)

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

P(t)

V(s)

P(s)

P(s) V(s) P(t) V(t)

Red Squares denote : critical section

Green lines denote : possible paths

Purple X denote: dead lock

Initially: s =1 , t= 0

B) Yes, It always deadlocks because since t = 0 from the start the sema does not let anything in.

C) If we were to initialize t = 1 then that would solve the problem.

D)

V(t)

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

P(t)

V(s)

P(s)

P(s) V(s) P(t) V(t)

**5)**

It cannot dead lock, first we notice that (a) is only used in thread 1 so we can look beyond that:

Thread1 Thread2

P(b) P(c)

V(b) P(b)

P(c) V(b)

V(c) V(c)

We can see that if Thread1 grabs (b) it will promptly release it, allowing Thread2 to continue and release (c) for Thread1. If Thread2 grabs b first it also releases right after allowing Thread1 to start and complete after Thread2 also releases (c).

**6)**

Initially a=1 b=1 c=1

A)

Thread1: (ab) and (ac)

Thread2: (cb)

Thread3: (ab)

B) Thread2 and Thread3 violate the ordering rule

C) Thead2: Thread3:

P(a) ; P(a);

P(b); V(a);

V(b); P(b);

V(a); P(c);

P(c); V(c);

V(c); V(b);

**7)**

A)they are necessary

B)not necessary

C)not necessary

**8)**

Mutex lock= 1, Mutex readers =n , Mutex hold = 1

Read:

P(hold)

P(readers)

if (readers == n-1 ) P(lock)

V(hold)

//read

V(readers)

if(readers == n) V(lock)

Write:

P(hold)

P(lock)

//write

V(lock)

V(hold)

**9)**

Files provided you can run it calling: make test