Real Time Group

מרכז להכשרות מקצועיות והשמה בתעשיית ההייטק

These are the methods, variables and definitions that you may use throughout home and class exercises :

Definitions:

#define INIFINITE_TIMEOUT	0xFFFFFFFF
#define POLICY_FIFO	0x1
#define POLICY_PRIORITY	0x2
#define BLOCK_IF_FAILE	0x1
#define ERROR_IF_FAILE	0x2
#define MAX_PRIORITY	50
#define MIN_PRIORITY	150

Variables:

Sem[100], SemA, SemB, SemC, SemD, TimeoutMilliSeconds, MsgQ[100], MsgQ1, MsgQ2, MsgQ3, MsgQ4, task1, task2, task3, task4, task[100]

Methods:

Mutex:

SemMCreate (&Sem, policy), SemMTake (&Sem, timeoutvalue), SemMGive (&Sem)

Inversion Safe Mutex

SemMPISafeCreate (&Sem), SemMPISafeTake (&Sem, timeoutvalue), SemPISafeGive(&Sem)

Binary Semaphores:

SemBCreate (&Sem, initialHoldingCount, policy), SemBTake (&Sem, timeoutvalue, shouldBlockIfFaile), SemBGive (&Sem)

Counting Semaphores:

SemCCreate (&Sem, initialHoldingCount, maximumAllowedHoldings, policy), SemCTake (&Sem, timeoutvalue, shouldBlockIfFaile), SemCGive (&Sem)

Message Queues:

MsgQCreate(&MsgQ, elementSize, NumOfElements, policy), MsgQSend (&MsgQ, inbuffer, timeoutvalue, shouldBlockIfFaile), MsgQReceive(&MsgQ, outBuffer, timeoutvalue)

Tasks:

TaskCreate (&Task, priority), TaskDelay (timeoutvalue), TaskSetPriority (&Task, &NewPriority) TaskSuspend(&Task), TaskResume(&Task), TaskLock(), TaskUnlock()

Interrupts:

DisableIntr (), EnableIntr ()

Busy wait: Hold_CPU_For (time-units)

Please assume for the purpose of the questions that non-blocking operations run at 0 time units.

Real Time Group

מרכז להכשרות מקצועיות והשמה בתעשיית ההייטק

1-

You have a system that consists of 4 tasks: Task1, Task2, Task3, Task4. Task1 is the first running task and its priority is 50.

Please draw a timeline of the following system:

```
Task1 ()
{
        TaskCreate (&T2, 100);
        TaskCreate (&T3, 120);
        TaskCreate (&T4, 150);

        Hold_CPU_For (500)
}

Task2 ()
{
        Hold_CPU_For (500)
}

Task3 ()
{
        Hold_CPU_For (250)
        TaskDelay (500)
        Hold_CPU_For (250)
}

Task4 ()
{
        while (1);
}
```

2-

You have a system that consists of 4 tasks: Task1, Task2, Task3, Task4. Task1 is the first running task and its priority is 50. Task2 and Task3 consist of the same function.

Note:

Please assume that the auto increment operation on the variable a, is atomic. Assume that the compiler has been instructed to create assembly code that uses processor-locking primitives.

```
GLOBAL int a = 0;
Task1 ()
{
        TaskCreate (&T2, 100);
        TaskCreate (&T3, 100);
        TaskCreate (&T4, 80);

}

Task2 ()
{
        int I;

        for (I=0; I < 100; I ++)
        {
                a ++;
        }
        while (1);
}

Task3 ()
{
        The same as Task2
}

Task4 ()
{
        a = 80;
}
```

2.1 – What will be the maximum value of a in a Preemptive-Priority system. Why?

2.2 – What will be the maximum value of a in a Round Robin system. Why?

2.3 – If Task4 would have been created with the priority 150, what will be the impact on the previous answers? Will Task4 be able to get into running state?

3-
You have a system that consists of 4 tasks: Task1, Task2, Task3, Task4. Task1 is the first running task and its priority is 150.
Please draw a timeline of the following system:

```
Task1 ()
{
        TaskCreate (&T2, 100); TaskCreate (&T3, 120);
        TaskCreate (&T4, 140);
        while (1);
}

Task2 ()
{
        Hold_CPU_For (500)
}

Task3 ()
{
        Hold_CPU_For (250) TaskDelay (500)
        Hold_CPU_For (250)
}

Task4 ()
{
        Hold_CPU_For (750)
}
```