UTAustinX: UT.RTBN.12.01x Realtime Bluetooth Networks

Help

Bookmarks

1. Introduction to RTOS

2. Thread Management

2.0. Objectives

2.1. Introduction to Real-time Operating Systems

2.2. Interrupts

2.3. Thread Management

2.4. Semaphores

2.5. Thread Synchronization

Quiz 2
Quiz

Lab 2) Thread Management
Lab

References

3. Time Management

4. Real-time Systems

5. File Systems

6. Bluetooth Low Energy

2. Thread Management > Quiz 2 > Quiz 2

Quiz 2

Bookmark this page

Definitions

2/2 points (graded)

Please match the following terms with the letter of their appropriate definitions.

Thread

C ✓

A. A guarantee to meet all deadlines

Jitter

D ✓

B. Hardware/software triggered software action

Real time

A ✓

C. Either the execution of software or the action caused by software execution

Interrupt

B ✓

D. The difference between desired time a task is supposed to run and the actual time it is run

Callback

E ✓

E. A mechanism that uses function pointers

Submit

Glossary

Discussion Board

Schedulers

2/2 points (graded)

Please match the following terms with the letter of their appropriate definitions.

Round robin

B

✓

Priority

E

✓

Exponential queue

A

✓

Rate Monotonic

C

✓

Cooperative

D

✓

A. A dynamic scheduler that shifts importance depending on if the thread ran to the completition of its time slice.

B. Run the ready threads in circular fashion, giving each the same amount of time to execute

C. Assign importance according to these periods with more frequent tasks having higher importance.

D. Threads themselves decide when to stop running

E. Run the most important ready threads first, running less important threads only if there are no important threads ready

Submit

SysTIck Interrupts

2/2 points (graded)

Assume the bus clock is 50 MHz or 20 ns. You wish to configure SysTick to interrupt every 10ms.

What value should you write into STCTRL?

7 ✓

7

What value should you write into RELOAD?

499999 ✓

499999

What value should you set into PRIMASK?

0 ✓

0

Submit

Bug or no bug

2/2 points (graded)

We can use semaphores to limit access to resources. In the following example both threads need access to a printer and an SPI port. The binary semaphore **sPrint** provides mutual exclusive access to the printer and the binary semaphore **sSPI** provides mutual exclusive access to the SPI port. Consider the following scenario to see if it has any bugs.

Thread 1

```
bwait(&sPrint);
bwait(&sSPI);
OutSPI(4);
printf("Hasta luego");
OutSPI(6);
bsignal(&sPrint);
bsignal(&sSPI);
```

Thread 2

```
bwait(&sSPI);
bwait(&sPrint);
OutSPI(5);
printf("tchau");
OutSPI(7);
bsignal(&sSPI);
bsignal(&sPrint);
```

Does the above scenario have bugs?

○ Yes there is a bug, the two **OutSPI** should be together

○ No bug per se, but the semaphores are not needed

● Yes there is a bug, this has a possible deadlock ✓

○ Yes there is a bug, there are two semaphores and three outputs, one more semaphore is needed

○ Yes there is a bug, the order of the two signals needs to be switched

○ No, this code has no bugs

Submit

Three tasks

1/1 point (graded)

You have three tasks. Task 1 takes a maximum of 1 ms to execute and runs every 10 ms. Task 2 takes a maximum of 0.5 ms to execute and runs every 1 ms. Task 3 takes a maximum of 1 ms to execute and runs every 100 ms.

Is there a possible scheduling algorithm for these three tasks?

○ No, there is not enough time

○ Maybe, but we need to know more information

● Yes ✓

○ There is no way to tell without running it

Submit

Four tasks

1/1 point (graded)

You have four tasks. Task 1 takes a maximum of 1 ms to execute and runs every 5 ms. Task 2 takes a maximum of 0.5 ms to execute and runs every 2 ms. Task 3 takes a maximum of 1 ms to execute and runs every 20 ms. Task 4 takes a maximum of 6 ms to execute and runs every 10 ms.

Is there a possible scheduling algorithm for these three tasks?

○ No, there is not enough time

○ Maybe, but we need to know more information

● Yes ✓

○ There is no way to tell without running it

Answer

Correct: think about the total time each tasks runs in 1 second

Submit