What is round-robin scheduling?

Asked 16 years, 3 months ago Modified 4 years, 8 months ago

Viewed 26k times

15

In a multitasking operating system context, sometimes you hear the term round-robin scheduling. What does it refer to?
What other kind of scheduling is there?

operating-system computer-science scheduler

Share

Improve this question

Follow

edited Jul 24, 2015 at 4:58

Premraj

74.4k • 26 • 243 • 184

asked Sep 17, 2008 at 2:43

Benoit

38.9k • 24 • 85 • 117

7 Answers

Sorted by: Highest score (default)

25

Round Robin Scheduling

If you are a host in a party of 100 guests, round-robin scheduling would mean that you spend 1 minute (a fixed amount) per guest. You go through each guest one-by-

one, and after 100 minutes, you would have spent 1 minute with each guest. More on Wikipedia.

There are many other types of scheduling, such as priority-based (i.e. most important people first), first-come-first-serve, earliest-deadline-first (i.e. person leaving earliest first), etc. You can start off by googling for scheduling algorithms or check out scheduling at Wikipedia

Share Improve this answer

Follow

answered Sep 17, 2008 at 2:47

Swati

52.6k ● 4 ● 40 ● 54

7

Timeslicing is inherent to any round-robin scheduling system in practice, AFAIK.

I disagree with InSciTek Jeff's implication that the following *is* round-robin scheduling:

> That is, each task at the same priority in the round-robin rotation can be allowed to run until they reach a resource blocking condition before yeilding to the next task in the rotation.

I do not see how this could be considered round-robin. This is actually preemptive scheduling. However, it is possible to have a scheduling algorithm which has elements of both round-robin and preemptive scheduling, which VxWorks does if round-robin scheduling and

preemption are both enabled (round-robin is disabled by default). The way to enable round-robin scheduling is to provide a non-zero value in *kernelTimeSlice*.

I do agree with this statement:

> Therefore, while timeslicing based scheduling implies round-robin scheduling, round-robin scheduling does not require equal time based timeslicing.

You are right that it doesn't require equal time. Preemption can muck with that. And actually in VxWorks, if a task is preempted during round-robin scheduling, when the task gets control again it will execute for the rest of the time it was allocated.

Edit directed at InSciTek Jeff (I don't have comment privileges) Yes, I was referring to task locking/interrupt disabling, although I obviously didn't express that very well. You preempted me (ha!) with your second comment. I hope to debate the more salient point, that you believe round-robin scheduling can exist without time slicing. Or did you just mean equal time based time slicing? I disagree with the former, but agree with the latter. I am eager to learn. Thanks.

Edit2 directed at Jeff:

> Round-robin can exist without timeslicing. That is exactly what happens in VxWorks when

> kernelTimeSlice is disabled (zero).

I disagree with this statement. See this document section 2.2.3 with the heading Round-Robin Scheduling.

> Round-robin scheduling uses time slicing to achieve fair allocation of the CPU to all tasks with the same priority. Each task, in a group of tasks with the same priority, executes for a defined interval or time slice. Round-robin scheduling is enabled by calling kernelTimeSlice( ), which takes a parameter for a time slice, or interval. [...] If round-robin scheduling is enabled, and preemption is enabled for the executing task, the system tick handler increments the task's time-slice count.

Timeslicing is inherent in round-robin scheduling. Otherwise you are relying on a task to give up CPU control, which round-robin scheduling is intended to solve.

edited Sep 19, 2008 at 15:59

answered Sep 17, 2008 at 15:03

unwieldy
197 • 3

You can't turn off preemption in VxWorks. It is inherent because if a higher priority task becomes runnable, it will PREEMPT the lower priority thread. Preemption is not tied to timeslicing, it involves any situation where a thread can be interrupted without explicitly having to yield the CPU. – Tall Jeff Sep 18, 2008 at 4:11

Well, to be clear - in case somebody gets very literal - preemption is inherent in VxWorks....assuming you don't disable interrupts and/or lock the scheduler. – Tall Jeff Sep 18, 2008 at 4:20

RE Edit: Round-robin can exist without timeslicing. That is exactly what happens in VxWorks when kernelTimeSlice is disabled (zero). – Tall Jeff Sep 19, 2008 at 2:09

RE Edit: To me the preemptive in preemptive scheduling or preemptive multi-tasking just refers to the fact/feature of the kernel in that it facilitates asynchronous deschedling of the thread without the thread having to call a kernel API method to allow a task switch to occur. – Tall Jeff Sep 19, 2008 at 2:12

RE Edit: In non-preemptive systems, for task switches to occur, the thread MUST call a kernel blocking primitive for a switch to occur or it must explicitly make some sort of task_yield() call to cause the switch. – Tall Jeff Sep 19, 2008 at 2:17

3

The answers here and even the Wikipedia article describe round-robin scheduling to inherently include periodic timeslicing. While this is very common, I believe that Round-Robin scheduling and timeslicing are ***not*** exactly the same thing. Certainly, for timeslicing to make sense, round-robin schedling is implied when rotating to

each task, however you can do round-robin scheduling without having timeslicing. That is, each task at the same priority in the round-robin rotation can be allowed to run until they reach a resource block condition and only then having the next task in the rotation run. In other words, when equal priority tasks exist, the reschedling points are ***not*** time pre-emptive.

The above idea is actually realized specifically in the case of Wind River's VxWorks kernel. Within their priority scheme, tasks of each priority run round robin but do not timeslice without specifically enabling that feature in the kernel. The reason for this flexibility is to avoid the overhead of timeslicing tasks that are already known to run into a block within a well bounded time.

Therefore, while timeslicing based scheduling implies round-robin scheduling, round-robin scheduling does not require equal time based timeslicing.

Share Improve this answer Follow

edited Sep 18, 2008 at 4:17

answered Sep 17, 2008 at 13:40

Tall Jeff

9,984 ● 7 ● 46 ● 61

1

An opinion. It seems that we are intertwining two mechanisms into one. Assuming only the OP's original

assertion "In a multitasking operating system context" then

1 - A round robin scheduler always schedules the next item in a circular queue.

2 - How the scheduler regains control to perform the scheduling is separate and unrelated.

I don't disagree that the most prevalent method for 2 is time-slicing / yield waiting for resource, but as has been noted there are others. If I am not mistaken the first Mac's didn't utilize time-slicing, they used voluntary yield / yield waiting for resource (20+ year old brain cells can be wrong sometimes;).

Share Improve this answer

Follow

answered May 23, 2010 at 12:35

dbasnett

11.8k ● 2 ● 26 ● 35

Round robin is a simple scheduling algorithm where time is divided evenly among jobs without priority.

For example - if you have 5 processes running - each process will be allowed to run for 1/5 a unit of time before another process is allowed to run. Round robin is typically easy to implement in an OS.

0

Share Improve this answer

Follow

answered Sep 17, 2008 at 2:48

mbowcock

11 ● 3

Actaully, you are getting confused with Preemptive scheduling and Round robin. Infact RR is part of Preemptive scheduling.

0

Share Improve this answer

Follow

answered Apr 9, 2010 at 7:12

Ashish

11

It isn't. RR is a method of distributing execution time among tasks that are runnable. Preëmptive scheduling characterizes the fact that a task can be interrupted at arbitrary points. RR can be used with coöperative scheduling too, and there are other alternatives to RR (e.g., priority-based).
– Donal Fellows May 23, 2010 at 12:47

It is preemptive at time quantum partially according to William Stallings. – Ankur Sinha Jul 2, 2013 at 8:08

0

Round Robin scheduling is based on time sharing also known as quantum (max time given by CPU to any process in one go). There are multiple processes(which require different time to complete aka burst time) in a queue and CPU has to process them all so it keeps switching between processes to give every process equal time based on the quantum value. This type of scheduling is known as Round Robin scheduling. Checkout this simple video to understand round robin scheduling easily: https://www.youtube.com/watch?v=9hw-_qJ55K4

Share Improve this answer

Follow

answered Feb 16, 2017 at 14:41

tanmay sakpal

1 ● 1