Quora uses cookies to improve your experience. Read More

What is a tickless kernel?

Answer Follow · 81 Request 1

2 Answers Best

Robert Love, I hack on Linux & worked on Android.
Updated Sep 25, 2013

To understand a tickless kernel, we first have to understand how a kernel usually functions. Numerous functions of an operating system need to be performed periodically—updating the time, decrementing the currently-running process's timeslice, expiring timers, and so on. To facilitate these periodic operations, systems provide a hardware component called the *programmable interrupt timer* (*PIT*), also known as the *system timer* or just the *timer*. The timer allows the operating system to schedule an interrupt with some fixed granularity—say, every 10 milliseconds. The kernel can then register a function, known as the *timer interrupt handler*, to respond to this timer interrupt and perform the aforementioned time-related housekeeping. As the timer hits at a fixed and known interval, every execution of the timer interrupt handler allows the kernel to know that, say, 10 milliseconds have elapsed. Every hit of the timer is known as a *tick*.

There are two major downsides to this model. The first is that deciding the interval of the timer interrupt is a tradeoff: Too low a frequency and the granularity of the kernel's time keeping is suboptimal. For example, if your timer interrupt hits every 10 milliseconds and the timer interrupt is how you keep track of process timeslice, then you have a floor of 10 milliseconds in the accuracy of your process scheduler. Conversely, too high a frequency and you waste a lot of processor time processing the timer interrupt when it isn't needed.

The second tradeoff is power consumption. On a mobile device that achieves its long uptime through aggressive power management, the timer tick can cause 100s or even 1000s of wake ups a second, depending on its frequency. That'll eat through your battery, as your device is never truly idle.

Enter "tickless" kernels. In such designs, there is no fixed timer tick. Instead, the kernel schedules the next timer tick in response to its next event. Perhaps a better name is a "dynamic tick" kernel. If the running process has 18 milliseconds of timeslice left, then the kernel can dynamically schedule the timer interrupt for 18 milliseconds from now. This allows the system to negate the tradeoffs of a fixed interval: High frequency when you need the granularity, low frequency when you don't. Moreover, if the system is idle, there is no reason to schedule a periodic timer tick at all. The system can go truly idle, vastly improving battery life.

The cons are complexity. A tick-based system is significantly simpler than a tickless one. As a programmer, it is much easier to develop kernel code where you know there is some fixed frequency to the timer interrupt and you can rely on the tick to always hit at that interval. On a system such as Linux that supports tickless operation you are more-or-less paying the cost whether or not you enable the feature. The major cost to turning a tickless system on is computation of the dynamic tick and the rescheduling of timers. For most workloads, this is negligible.

You can read more about Linux's timer interrupt handler and tickless implementation at Documentation/timers/NO_HZ.txt in the kernel source or *Chapter 11* in *Linux Kernel Development* (disclaimer: by me).

62.8K views · View 605 upvotes · View 10 shares

605 10 8

8 comments from Ben Fraser and more

Peter Teoh
Answered Nov 28, 2015

6

Related questions

Why is colonel pronounced kernel?

What makes up the kernel?

What is NT Kernel & System?

What is the difference between kernel micro, kernel, and hypervisor?

Which is the best stock, Kernel or Custom Kernel?

Which is the most widely used kernel, the Gaussian kernel or the Chi-Square kernel?

Add question