soft vs. hard interrupt handle timing in Linux

I'm converting some software from using a physical HW device, to a total software simulation, and right now I'm looking at the interrupt handling portion.

When this code was driving physical HW, it was requesting an IRQ via `request_irq()` , now that I no longer have a physical device I was going to change the code to use the "softirq" methods.

One concerning point, I see that softirqs are actually just events that have to wait until it is called by the scheduler; whereas with a hardware interrupt causes the immediate interruption of the running activity. This implies to me that by converting my code to use softirqs will cause scheduling delays that were not present in the original code.

Is my understanding correct?
Is there anyway I can register my softirq to interrupt immediately like a HW irq would?
If not is there anyway to pick a free HW irq number and keep using `request_irq()` and "trick" it into thinking that I'm running a HW device?

c linux scheduled-tasks interrupt irq

asked Oct 10 '12 at 14:05

Mike
17.6k 8 35 91

1 There will almost always be timing differences when moving from true hardware to a software simulation - this holds true with or without interrupt delivery timing differences. – mah Oct 10 '12 at 14:12

@mah - true, but the goal is to remove as much of that as possible. In this case I **want** the other software entities to be impacted (interrupted). If not for timing per say, I'd like them to be forced to be stopped while the IRQ is handled. – Mike Oct 10 '12 at 14:16

@Mike: Please add more information about what exactly will be triggering interrupts in new scenario and what is your software doing. If you're going to trigger them from userland, maybe you shouldn't use softirqs at all, but switch to more appropriate API for a device. – dpc.ucore.info Oct 10 '12 at 17:05

2 Answers

You could cause an exception either in the user-mode code or in the kernel-mode code (via an additional system call or a driver) and have a dedicated exception handler in the kernel to transform this exception into the simulated interrupt. Exceptions are typically handled immediately, at least by the CPU, similar to external hardware interrupts. I don't know enough of Linux internals to tell you exactly how to accomplish this, but it must be doable. I'm certain, it would work on x86.

answered Oct 10 '12 at 14:24

Alexey Frunze
39k 7 25 64

That's actually a great idea, I should be able to accomplish that, just didn't consider it as an option. Thanks! – Mike Oct 10 '12 at 15:28

I'm not totally confident in the field of Real Time computing but this sounds like a good use case of it. Real Time computing assures that your code get's executed in a guaranteed time frame. There is a Linux Real-Time Kernel available which sadly lacks funding. If you still need guaranteed execution of your softirq you should look into it. I know it's a pretty question but maybe it's also useful for others.

answered Jan 12 at 17:05

JaVaEs
13 3