

## Re: getting timestamp of last interrupt?

**Source:** <http://linux.derkeiler.com/Mailing-Lists/Kernel/2003-10/1327.html>

---

**From:** Richard B. Johnson ([root\\_at\\_chaos.analogic.com](mailto:root_at_chaos.analogic.com))

**Date:** 10/06/03

Date: Mon, 6 Oct 2003 15:33:22 -0400 (EDT)  
To: Hans-Georg Thien <[1682-600@onlinehome.de](mailto:1682-600@onlinehome.de)>

On Mon, 6 Oct 2003, Hans-Georg Thien wrote:

> *Richard B. Johnson wrote:*

>

> > *On Mon, 6 Oct 2003, Hans-Georg Thien wrote:*

> >

> >>

> >>[...]

> >>*I'm writing a kernel mode device driver (mouse).*

> >>

> >>*In that device driver I need the timestamp of the last event for another*

> >>*kernel mode device (keyboard).*

> >>

> >>*I do not care if that timestamp is in jiffies or in gettimeofday()*

> >>*format or whatever format does exist in the world. I am absolutely sure*

> >>*I can convert it somehow to fit my needs.*

> >>

> >>*But since it is a kernel mode driver it can not -AFAIK- use the signal()*

> >>*syscall.*

> >>

> >>*-Hans*

> >

> >

> > *Then it gets real simple. Just use jiffies, if you can stand the [...]*

> *I fear that there is still some miss-understanding. Jiffies are totally*

> *OK for me. I can use them without any conversion.*

>

> *I'll try to formulate the problem with some other words:*

>

> *I hope that there is something like a "jiffie-counter" for the*

> *keyboard driver, that stores the actual jiffies value whenever a*

> *keyboard interrupt occurs.*

>

Well the keyboard driver and the mouse driver are entirely different devices.

## Linux–Kernel: Re: getting timestamp of last interrupt?

The keyboard has a built–in CPU that generates scan–codes for every key–press/key–release. It also performs auto–repeat. The mouse generates mouse data at each interrupting event. This data represents direction and three key events. Wheel mouse have may have additional data, I haven't looked at them. They are not related in any way.

```
> I hope too, that there is a way to query that "jiffie–counter" from
> another kernel driver, so that I can write something like
>
>
> mymouse_module.c
>
> ...
> void mouse_event(){
>
> // get the current time in jiffies
> int now=jiffies;
>
> // get the jiffie value of the last kbd event
> int last_kbd_event= ???; // ... but how to do that ...
>
> if((now – last_kbd_event) > delay) {
> do_some_very_smart_things();
> }
> }
> ...
>
```

Now this pseudo–code shows a "last\_kbd\_event", not a mouse–event as shown in:

```
> >>I'm writing a kernel mode device driver (mouse).
```

... your words, not mine.

I presume that you are replacing the existing mouse–driver. If so, your mouse–buffer, i.e., the place you put the mouse–event movement–codes can be something like:

```
struct mouse_event {
    unsigned long time;
    int mouse_code;
}
```

You allocate a buffer of this type and, during each interrupt, you put both the jiffie–time and the mouse code into your buffer.

If you are not replacing the existing mouse driver, then you need to share its interrupt with your new module. The new module records the time–stamp of each of the interrupts, only.

Re: getting timestamp of last interrupt?

## Linux-Kernel: Re: getting timestamp of last interrupt?

Every bit of mouse-data was obtained as a result of an interrupt. Using that knowledge, you should be able to correlate a time-stamp to mouse-data (clear your time-stamp buffer when no mouse data are available).

If you are using the keyboard, not the mouse, then the same things apply.

If you just want to patch the existing keyboard or mouse ISR to save a time-stamp in your module code, you need to make the built-in keyboard or mouse ISR code call a function by pointer. This pointer must be initialized to point to a stub that simply returns. You need to export this symbol so it can be found by your module.

When your module is installed, it saves the value in that pointer. It then changes the pointer value to the address of your routine. It needs to do this under a spin-lock.

When your module is un-installed, it needs to restore the previous (saved) value of that pointer.

Whatever code you make that pointer point-to, must be interrupt-safe. It can get the jiffie-count and put it into a buffer, then return.

Cheers,

Dick Johnson

Penguin : Linux version 2.4.22 on an i686 machine (797.90 BogoMips).

Note 96.31% of all statistics are fiction.

-

To unsubscribe from this list: send the line "unsubscribe linux-kernel" in the body of a message to [majordomo@vger.kernel.org](mailto:majordomo@vger.kernel.org)

More majordomo info at <http://vger.kernel.org/majordomo-info.html>

Please read the FAQ at <http://www.tux.org/lkml/>