

# Re: Serial port blues

---

*Source:* <http://linux.derkeiler.com/Mailing-Lists/Kernel/2007-01/msg05218.html>

---

- *From:* Willy Tarreau <[w@xxxxxx](mailto:w@xxxxxx)>
  - *Date:* Sun, 21 Jan 2007 08:05:57 +0100
- 

On Sun, Jan 21, 2007 at 12:54:56AM -0500, Theodore Tso wrote:

On Sat, Jan 20, 2007 at 06:36:44PM +0100, Willy Tarreau wrote:

On Fri, Jan 19, 2007 at 03:37:34PM -0600, Joe Barr wrote:

I'm forwarding this post by the author of a great little program for digital amateur radio on Linux, because I'm curious whether or not the problem he is seeing can be resolved outside the kernel.

At least, I see one wrong claim and one unexplored track in his report. The wrong claim : the serial port can only be controlled by the kernel. It is totally wrong for true serial ports. If he does not want to use `ioctl()`, then he can directly program the I/O port.

There's more wrong with his claim than just that. Another wrong claim is that it's caused by the Linux kernel not treating `ioctl` requests with high priority. Of course that's nonsense. It might be the case if we were using brain-damaged messaging-passing approach like what Andrew Tenenbaum is proposing with Minix 3.1, but in Linux, the serial port DTR/CTS lines are toggled as soon as the userspace executes the `ioctl`.

Damn you're right. It shocked me too and I know I was missing something when replying but I did not remember what.

The real issue is when does the userspace program get a chance to run. He's using the `select()` system call, which only guarantees accuracy up to the granularity of the system clock. Given that he's reporting a jitter of between 0 and 4ms, I'm guessing that he's running with a system clock tick of 250HZ (since  $1/250 == 4\text{ms}$  ).

## Re: Serial port blues

Yes, that's what I thought too. In the past, I've been having better resolution with `select()` and real-time scheduling, but I cannot reliably reproduce it, even on SMP. I remember nothing was running at all on the machine (not even X) and that can make an important difference. But as you say, there will be no guarantee of better accuracy anyway.

So if he wants accuracy greater than that, there are a couple of things he can do. One is to recompile his kernel with `HZ=1000`. That will give him accuracy up to 1ms or so. If he needs better than 1ms granularity, there are two options. One is use `sched_setscheduler()` to enable posix soft-realtime, and then calibrate a busy loop. This will of course burn power and completely busy out one CPU, so if he needs to run CW continuously this probably isn't a great solution. On an SMP system it might work, although it is obviously a huge kludge.

Hmmm the busy loop is dirty as hell, even on SMP, but it works ;-) I remember it was possible to reprogram the RTC to interrupt at 8192 Hz. If the task is running with real time prio, it should get this accuracy, or am I mistaken ?

The other choice would be to install Ingo's `-rt` patches (see <http://rt.wiki.kernel.org> for more information), and then use the Posix high-resolution timer API's (i.e., `timer_create`, et. al). Make sure you enable `CONFIG_HIGH_RES_TIMERS` after you apply the patch. It would also be a good idea to set a real-time scheduling priority for the application, to make sure that when the timer goes off, the process doesn't get preempted by some background cron job.

Now he must be careful about avoiding busy loops in the rest of the program, or he will have to use the reset button.

An easy way of dealing with this is to have an `sshd` running an alternative port running at a nice high priority (say, prio 95 or so). That way, if you screw up, you can always login remotely and kill the offending program.

There is also a RT Watchdog program which can be found on [rt.wiki.kernel.org](http://rt.wiki.kernel.org) which can be used to recover from runaway real-time processes without needing to hit the reset button.

Thanks for those hints, I've been used to play with the reset button, at least it has forced me to double check my code before running it :-)

Re: Serial port blues

Finally, please feel free to direct your amateur radio friend to the  
linux-rt-users@xxxxxxxxxxxxxxxxx There are plenty of folks there who  
would be very happy to help him out.

73 de Ted, N1ZSU

Regards,  
Willy

—

To unsubscribe from this list: send the line "unsubscribe linux-kernel" in  
the body of a message to majordomo@xxxxxxxxxxxxxxxxx

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

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