

Re: splxxx level?

Source: <http://unix.derkeiler.com/Mailing-Lists/FreeBSD/arch/2004-08/0102.html>

From: M. Warner Losh (*imp_at_bsdimp.com*)

Date: 08/30/04

Date: Mon, 30 Aug 2004 10:26:06 -0600 (MDT)

To: scottl@freebsd.org

In message: <41334C3B.4070101@freebsd.org>

Scott Long <scottl@freebsd.org> writes:

: Sam wrote:

:

:> Hello –

:>

:> I'm almost to testing on my AoE driver for 4.x and have

:> a question about interrupt priority levels.

:>

:> There are currently three entry points into the driver:

:>

:> a) strategy routine

:> b) network frame reception routine

:> c) timer rexmit routine

:>

:> Any of the three can diddle with the device structure

:> and thusly I need to ensure they're not running simultaneously.

:> For example, the network reception can cause a buf to be completed

:> and the rexmit timer can cause a buf to be failed.

:>

:> So, what kind of contexts are the callout, strategy, and

:> network soft interrupt called in? Which splxxx will give

:> one of them exclusive access to whatever they need?

:>

:> Just as a reality check — I am thinking about this correct, right?

:>

:> Cheers,

:>

:> Sam

:>

:

: With 4.x, only one CPU can be in the kernel at a time. You won't have

: to worry about multiple processes trying to get into strategy at the

: same time and whatnot. However, you can be preempted by your interrupt

: handler or by a timeout or by a software interrupt like the netisr. I

: don't remember if your driver is for a specific piece of hardware or if

: it's a generic layer that sits in between the network interface and the

freebsd-arch: Re: splxxx level?

: block layer. If it's for dedicated hardware then you'll need to define
: a interrupt type in bus_setup_intr() and use that type for the spl
: (i.e. INTR_TYPE_NET translates to splnet(), INTR_TYPE_BIO translates to
: splbio(), etc).
:
: The safe way to go is to protect all of your critical code sections with
: the appropriate spl level regardless. spls are very cheap and can be
: set/reset from an interrupt context so there is little penalty in using
: them liberally at first and then narrowing them down later. Just make
: sure that you don't leak an spl references, and don't hold an spl for so
: long that it creates priority inversions. Since the only interrupts and
: timeouts that you'll likely be dealing with are network related,
: splnet() is probably the right one to use.

splimp() is what you want to use, not splnet(). Yes, this is confusing, but it appears to be what all the other network drivers use. None of them are using splnet() that I could find. splimp() is also used by the mbuf routines to protect mbuf operations.

splnet() is a list of the software interrupts that are network drivers.

splimp() is splnet() plus the hardware interrupts, so is more appropriate to block things called from the driver. Especially one that's described as having timeouts. If it is a network driver, you might consider using the timeout functionality in the net stack as opposed to the callout functions. This makes it possible to have almost the entire driver w/o doing any spls (most of the network drivers in 4 don't do spl at all, except for entry points that are outside the scope of the network/interrupt entry points, eg device_suspend).

Warner

freebsd-arch@freebsd.org mailing list

<http://lists.freebsd.org/mailman/listinfo/freebsd-arch>

To unsubscribe, send any mail to "freebsd-arch-unsubscribe@freebsd.org"