

Re: API change for bus_dma

Source: <http://unix.derkeiler.com/Mailing-Lists/FreeBSD/arch/2003-06/0339.html>

From: Scott Long (scottl_at_freebsd.org)

Date: 06/29/03

Date: Sat, 28 Jun 2003 20:03:20 -0600
To: "Justin T. Gibbs" <gibbs@scsiguy.com>

Justin T. Gibbs wrote:

>> *Ok, after many semi-private discussions, how about this:*

>

>

> *There is only one problem with this strategy. The original idea*
> *of using a mutex allowed the busdma API to use that same mutex as*
> *the strategy for locking the fields of the tag, dmamap, etc. In*
> *other-words, the agreement would have been that the caller always*
> *has the lock held before calling into bus dma, so that bus dma*
> *only has to grab additional locks to protect data shared with*
> *other clients. For this to work in the more general scheme, you*
> *would have to register "acquire lock"/"release lock" functions in*
> *the tag since locking within the callback does not allow for the*
> *protection of the tag or dmamap fields in the deferred case (they*
> *would only be protected *during* the callback).*

>

> *Again, what we want to achieve is as few lock acquires and releases*
> *in the common case as possible. For architectures like x86, the only*
> *data structure that needs to be locked for the common case of no deferral*
> *and no bounce page allocations is the tag (it will soon hold the S/G list*
> *passed to the callback). Other implementations may need to acquire other*
> *locks, but using the client's lock still removes one lock acquire and*
> *release in each invocation that is not deferred.*

>

> --

> *Justin*

>

>

This is becoming wonderfully complex. What is the purpose of storing the S/G list in the tag? Are we going to enforce a 1:1 relationship between tags and maps? That would really suck for the aac(4) driver.

Scott

freebsd-arch@freebsd.org mailing list

freebsd-arch: Re: API change for bus_dma

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

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