

Re: Policy on rebooting?

Source: <http://coding.derkeiler.com/Archive/General/comp.arch.embedded/2004-10/0587.html>

From: Michael N. Moran (mike_at_mnmoran.org)

Date: 10/07/04

Date: Thu, 07 Oct 2004 07:38:43 -0400

D. Zimmerman wrote:

> *Paul Keinanen wrote:*

>

>> *On Wed, 6 Oct 2004 11:16:58 +0200, "mk" <REVERSE_lp.pw@myzskm.REMOVE>*

>> *wrote:*

>>

>>

>>> *Why is dynamic memory allocation bad idea?*

>>

>>

>>

>> *Fragmentation.*

>>

>

> *This also depends on the operating system being used. Most real time*

> *OS's, which are often used for embedded, will only allocate fixed size*

> *memory blocks and will reclaim the entire block. This prevents the*

> *problem Paul described.*

This does not **prevent** the problem, it only limits/slows the problem if the application has a particular pattern of dynamic memory usage.

Imagine a limited supply of large fixed size blocks. If these blocks are allocated and released and then subsequently allocated and fragmented to fulfill the needs of another part of the system requiring a smaller block size (of which there are no more), then eventually such an allocation system will fail as well.

Of course, we're talking about general malloc/free new/delete here, not private memory pools, which are pre-allocated for a particular part of the system with fixed sized blocks.

What's more is that block memory allocation schemes reduce the "memory efficiency" advantages of dynamic memory allocation by wasting the unused portion of each allocated block.

BTW, allocation is not the issue. If memory is never freed, then fragmentation will not happen. Of course, then its not

comp.arch.embedded: Re: Policy on rebooting?

really dynamic.

--

Michael N. Moran (h) 770 516 7918
5009 Old Field Ct. (c) 678 521 5460
Kennesaw, GA, USA 30144 <http://mnmoran.org>
"... abstractions save us time working, but they don't
save us time learning."
Joel Spolsky, The Law of Leaky Abstractions
The Beatles were wrong: 1 & 1 & 1 is 1