

Re: Architectures with strict alignment?

Re: Architectures with strict alignment?

Source: <http://unix.derkeiler.com/Mailing-Lists/FreeBSD/hackers/2007-12/msg00287.html>

- *From:* Erik Trulsson <ertr1013@xxxxxxxxxxxxxx>
 - *Date:* Mon, 31 Dec 2007 14:04:17 +0100
-

On Mon, Dec 31, 2007 at 08:30:35PM +0800, Erich Dollansky wrote:

Hi,

Kostik Belousov wrote:

On Mon, Dec 31, 2007 at 05:38:43PM +0800, Erich Dollansky wrote:

Kostik Belousov wrote:

On Sat, Dec 29, 2007 at 01:12:04PM +0200,
Kostik Belousov wrote:

On Sat, Dec 29, 2007 at
12:14:11AM -0800, Kip
Macy wrote:

I.e., it seems that gcc does not feel too guilty
generating unaligned
half-word writes on i386. :(

this should not be a problem inside a cache line.

If the access goes accross two cache lines and the other cache
line is
not in the cache, it becomes real difficult.

I can't tell you what the hardware actually does in this case.

It should read the second affected cache line into the cache.
But what
happens if the second affected cache line is blocked by
another CPU while
the current CPU blocks the first cache line?

From the manual, 253668, 7.1.1:

I think we might get any half of the operation as a result.

Re: Architectures with strict alignment?

so, both CPUs are blocked as none can access the other cache line.

Is there really nothing in a normal PC to handle this?

No, the CPUs are not blocked, and the hardware handles it in a typical modern multi-core system (so the programmers can't get it wrong.)

If one CPU tries a write that crosses a cache-line boundary then what might happen (details vary among architectures) is that it will first get one cache-line, modify it, and write it back to memory. Then it will handle the other cache-line. This way you cannot get any deadlocks, since the CPU is waiting for at most one cache-line at a time, but you can get inconsistent results since the write is not atomic. (Another C