

Re: Pointing to high and low bytes of something

Source: http://coding.derkeiler.com/Archive/C_CPP/comp.lang.c/2003-11/3436.html

From: Chris Torek (nospam_at_elf.eng.bsdi.com)

Date: 11/20/03

Date: 19 Nov 2003 18:22:32 -0700

In article <3FBBBEF0.87B2475F@sun.com>

Eric Sosman <Eric.Sosman@Sun.COM> writes:

>... *I'm saying that (1) bit order is not detectable by any C
>construct I can imagine, (2) bit order is not detectable by any CPU
>instruction on a machine that lacks bit-level addressing,
>(3) by Occam's Razor, that which is undetectable is better
>omitted from discussion.*

Indeed. One can, however, bring up C's bitfield-in-structure construct:

```
struct bits { int a:1, b:1, c:1 /* fill in more */ ; };
```

which might **seem** to expose the hardware's bit order.

In fact, it does not — the bitfields are allocated by the compiler, and two different C compilers on the same hardware will sometimes use different bit orders.

Even for case (2), sometimes the CPUs themselves exhibit a split personality. The Motorola 680x0 series did this: the single-bit instructions that operate on D registers (e.g., BIT #3, D0) use the opposite order from the bitfield instructions (e.g., BFEXT).

[modern modem example, snipped]

> *The question: What is the "bit order" of the N bits
>encoded by one single BZZZ-to-BEEP transition in this scheme?
>Note that all N bits leave the transmitter encoded in one
>single event and arrive at the receiver the same way: they
>are simultaneous and indivisible — and I say the entire
>idea of "bit order" in such a situation is meaningless.*

Correct. Sequencing cannot arise unless there is a sequence. If there is no defined time or space division — if all is an indistinguishable, atomic lump — then the notion of "the part on the left" or "the part at the front of the queue" makes no sense.

comp.lang.c: Re: Pointing to high and low bytes of something

Of course, in C code, we (programmers) can take apart any byte (unsigned char) value in any way we like, using shifts and masks: val & 0x80, val & 0x01, val & 0x04, val & 0x02, val & 0x10, ... defines a bit order. But *we* have defined it, and thus we control the horizontal and the vertical. Only if you allow someone else to define the order -- say, by taking a two-byte object (one with sizeof obj == 2) and addressing it as two separate "unsigned char"s -- have you given up control; only then do you need to beg the one to whom you gave up that control: "pretty please, tell me the order *you* used, so that I may accomodate you". As Humpty Dumpty put it, the question is who is to be the master. :-)

--

In-Real-Life: Chris Torek, Wind River Systems
Salt Lake City, UT, USA (40°39.22'N, 111°50.29'W) +1 801 277 2603
email: forget about it <http://67.40.109.61/torek/index.html> (for the moment)
Reading email is like searching for food in the garbage, thanks to spammers.