

Re: "Interesting" C behaviours

Source: http://coding.derkeiler.com/Archive/C_CPP/comp.lang.c/2004-11/2934.html

From: Rennie deGraaf (ca.ucalgary.cpsc_at_degraaf)

Date: 11/27/04

Date: Sat, 27 Nov 2004 07:08:01 GMT

Malcolm wrote:

> "Rennie deGraaf" <ca.ucalgary.cpsc@degraaf> wrote

>>2. In C99, the expression 'a%b' where $a < 0$ and $b > 0$ will return a negative
>>result (K&R lets this be machine dependent).

>>

>

> That would probably be driven by architecture. The idea is that $a \% b$ would
> compile to a single machine instruction (on K and R's original platform).
> Just a guess, but I suspect this is the motivation.

It was architecture dependent in K&R, but C99 standardized it (see http://home.tiscalinet.ch/t_wolf/tw/c/c9x_changes.html#Semantics, section 25). I know that the x86 idiv instruction spits out negative residues, but is the fact that a common architecture does something unusual a reason to standardize a programming language on an unusual behaviour? It would make more sense to me to work around the architecture when it does something weird.

GCC, for instance, frequently doesn't even compile $a \% b$ to a single idiv instruction – it compiles it to a big mess of imul, shift, and leal instructions.

Rennie