

## Re: SET, CLR, ISSET in types.h for \_KERNEL builds

---

*Source:* <http://unix.derkeiler.com/Mailing-Lists/FreeBSD/arch/2006-06/msg00141.html>

---

- *From:* Bruce Evans <[bde@xxxxxxxxxxxx](mailto:bde@xxxxxxxxxxxx)>
  - *Date:* Wed, 28 Jun 2006 22:55:22 +1000 (EST)
- 

On Wed, 28 Jun 2006, Poul-Henning Kamp wrote:

In message <20060628094221.GA50978@xxxxxxxxxxxxxxxxxxxx>, Yar Tikhiy writes:

On Tue, Jun 27, 2006 at 01:58:17PM -0600, M. Warner Losh wrote:

NetBSD recently added SET, CLR, ISSET to sys/types.h  
(only if \_KERNEL  
is defined).

I see no types here.

As one of the people who have worked with the original /bin/sh source code, I have a slight allergic reaction to attempts to paste over the implementation language with macros like these.

```
Setting a bit in 'C' is spelled  
variable |= bit;  
not  
SET(variable, bit);
```

Only slightly?

In FreeBSD, these mistakes were only in kern/tty.c and in some usb files obtained from NetBSD and related to tty.c. In tty.c, they appear to be just to avoid adding a USL copyright. (tty.c was obfuscated between FreeBSD-1 and FreeBSD-2 by globally substituting "variable |= BIT" by "SET(variable, BIT)", etc.) I noticed that NetBSD started using these macros elsewhere many years ago. However, their use was still relatively limited in NetBSD a year ago (in kern, they were only in kern-fork.c (1), kern\_subr.c (2), kern\_systrace.c (many), sys\_process.c (a few), tty.c (many), tty\_pty.c (many), vfs\_bio.c (many) and vfs\_lookup.c

Re: SET, CLR, ISSET in types.h for \_KERNEL builds

(1); 211 lines altogether vs 1565 lines matching ' & ' and 27 lines matching the style bug '[A-Za-z]&[A-Za-z]'). It must have been in tty\_pty.c that I noticed them many years ago.

Higher order macros like roundup(), ispow2() are fine with me, because they implement something on top of the language and make the source code more compact and thus faster to read.

But all of the three proposed macros take up more space than the native language they obfuscate, what is the sense in that ?

They might be to hide the implementation of a set of flags as a bitmap, but they don't even do that.

Another problem with these macros is that a bitmap is more useful than a set of flags, but a much larger and uglier set of macros would be needed to give enough operations on bitmaps, and code that has been blindly translated from an integer-bitmap operation still assumes that the implementation is an integer-bitmap. E.g., in tty.c:

```
%%%
/*
 * delayed suspend (^Y)
 */
if (CCEQ(cc[VDSUSP], c) &&
    ISSET(lflag, IEXTEN | ISIG) == (IEXTEN | ISIG)) {
%%%
```

Bruce

---

freebsd-arch@xxxxxxxxxxxxx mailing list

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

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