

Arjan,

I can experiment with any constraints if you suggest which one.

>From our experiments with gcc, it compares asm strings (sic!!!) to find matches to be merged! Sigh...

Below are 2 programs which differ in one space in read_cr3_b() asm statement. The first one compiles incorrectly, while 2nd one - correctly.

My personal feeling is that comparing asm strings is simply a "misfeature".

```
----- cut -----
static inline unsigned long read_cr3_a(void)
{
    unsigned long cr3;
    asm("movq %%cr3,%0" : "=r" (cr3));
    return cr3;
}

static inline unsigned long read_cr3_b(void)
{
    unsigned long cr3;
    asm("movq %%cr3,%0" : "=r" (cr3));
    return cr3;
}

static inline void write_cr3(unsigned long val)
{
    asm volatile("movq %0,%%cr3" :: "r" (val) : "memory");
}

void main()
{
    unsigned long c;
    c = read_cr3_a();
    write_cr3(c | 0x80);
    c = read_cr3_b();
    write_cr3(c | 0x100);
}
----- cut -----
```

```
----- cut -----
static inline unsigned long read_cr3_a(void)
```

```

{
    unsigned long cr3;
    asm("movq %%cr3,%0" : "=r" (cr3));
    return cr3;
}

static inline unsigned long read_cr3_b(void)
{
    unsigned long cr3;
    asm("movq %%cr3,%0" : "=r" (cr3));
    return cr3;
}

static inline void write_cr3(unsigned long val)
{
    asm volatile("movq %0,%%cr3" :: "r" (val) : "memory");
}

void main()
{
    unsigned long c;
    c = read_cr3_a();
    write_cr3(c | 0x80);
    c = read_cr3_b();
    write_cr3(c | 0x100);
}
----- cut -----

```

Kirill

Arjan van de Ven wrote:

> On Tue, 02 Oct 2007 18:08:32 +0400

> Kirill Korotaev <dev@openvz.org> wrote:

>

>

>>Some gcc versions (I checked at least 4.1.1 from RHEL5 & 4.1.2 from

>>gentoo) can generate incorrect code with read_crX()/write_crX()

>>functions mix up, due to cached results of read_crX().

>>

>

>

> I'm not so sure volatile is the right answer, as compared to giving the

> asm more strict constraints....

>

> asm volatile tends to mean something else than "the result has

> changed"....

>

>