

## Re: user\_to\_phys() without mmap?

---

*Source:* <http://linux.derkeiler.com/Newsgroups/comp.os.linux.development.system/2007-07/msg00101.html>

---

- *From:* [phil-news-nospam@xxxxxxxx](mailto:phil-news-nospam@xxxxxxxx)
  - *Date:* 16 Jul 2007 12:28:11 GMT
- 

On Sun, 15 Jul 2007 00:20:45 +0200 Rainer Weikusat <rweikusat@xxxxxxxxxxxx> wrote:

| Neil Steiner <neil.steiner@xxxxxx> writes:

> I'm trying to use a framebuffer from user space (think plain video  
> ram, not some PCI or AGP video card), but mmap'ing from kernel space  
> into user space is causing large latencies and unsightly artifacts.  
> It appears that anything written into the mmap'ed user memory is in  
> fact being copied into kernel space.

|  
| Theoretically, this could probably be implemented with help of the  
| MMU and some 'weird' fault handling, but I don't think someone would  
| take the trouble, because it seems pretty useless. The /dev/mem driver  
| certainly doesn't do this. Only the MMU will interfere accesses to  
| virtual memory locations from user space, except for faults caused by  
| this accesses.

I tried to map the video buffer into user space some time ago and I did find that what was mapped was not the actual physical memory of the video card, but just some memory apparently allocated by a driver in the kernel which was then periodically copied from that memory into the real physical memory.

I speculated that the above was taking place because it was an attempt to port an old graphical program I had originally developed on DOS so it would run on Linux. It was a cellular life game program (simulation of cell growth and death activity). Under Linux 2.4 on a 400 MHz P-II it was getting speeds of not more than one generation every 2-3 seconds. Under DOS 3.3 on a 25 MHz i386, it was getting 16 generations per second. That's 32 to 48 times slower on a machine that could have been as much as 16 times faster. There was more than just copying data between buffers going on.

Directly mapping physical memory to a process can, of course, be very hazardous. Said program needs to be able to give up that mapping when needed, such as during virtual console switch. Presumably X windows is fully mapped to physical memory (it seems to be reasonably fast), but it is trusted to give up any such mappings when leaving X.

Still, it would be nice to have BOTH a way to easily get a direct mmap to physical video buffer memory, as well as an indirect mmap to a logical

Re: user\_to\_phys() without mmap?

buffer that can be kernel switched as needed. This could and should be done by having the user space virtual memory mapping go to the real device memory when that console is the currently selected one, and be mapped to a memory copy of the device memory at other times. That way, every virtual c