Importance of alignment even on x86 machines

devblogs.microsoft.com/oldnewthing/20040827-00

August 27, 2004

Raymond Chen

Sometimes unaligned memory access will hang the machine.

Some video cards do not let you access all the video memory at one go. Instead, you are given a window into which you can select which subset of video memory ("bank") you want to see. For example, the EGA video card had 256K of memory, split into four 64K banks. If you wanted to access memory in the first 64K, you had to select bank zero into the window, but if you wanted to access memory in the second 64K, then you had to select bank one.

Bank-switching makes memory access much more complicated, For example, if you want to copy a block of memory into bank-switched memory, you have to check when you are going to cross a bank boundary and break the copy up into pieces. If you are doing something that requires non-sequential access (say, drawing a diagonal line), you have to check when your line is going to cross into another bank.

To simplify matters, Windows 95 had a driver called VFLATD that made bank-switched memory look flat to the rest of the system. Flattening the bank-switched memory model was also crucial for DirectDraw support; in particular, the IDirectDrawSurface::Lock method gave you direct access to a (seemingly) flat expanse of video memory. For example, if the application wanted to see a 256K surface and accessed memory in the first 64K of memory, the VFLATD driver would select bank zero and map the 64K physical memory window into the first 64K of the virtual 256K memory window.

This worked great as long as everybody uses only aligned memory accesses. But if you access unaligned memory, you can send VFLATD into an infinite loop and hang the machine.

Suppose you make an unaligned memory access that straddles two banks. This memory access can never be satisfied. A page fault is taken on the lower portion of the unaligned access, and VFLATD maps the lower bank into memory. Then a page fault is taken on the higher portion of the unaligned access, and VFLATD now has to map the upper bank; this unmaps the lower bank, since the video card is bank-switched and only one bank can be mapped ata time. Now a page fault is taken on the lower portion, and the infinite loop continues.

Moral of the story: Keep those memory accesses aligned, even on the x86, which most people would consider to be one where it is “safe” to violate alignment rules.

Next time, another example of how misaligned data access can create bugs x86.

Raymond Chen

Follow