Why did Windows 95 keep window coordinates at multiples of 8?

devblogs.microsoft.com/oldnewthing/20210517-00

May 17, 2021

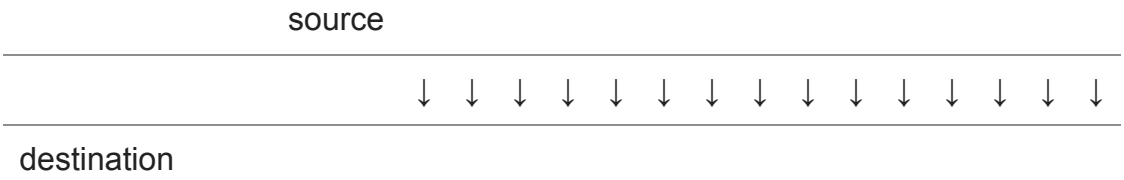
Raymond Chen

It is totally understandable but nonetheless feels weird to me that Windows 95 is considered to be retro-computing.

Okay, so why are window coordinates quantized at multiples of 8?

Because bit block transfers between coordinates that are *not* multiples of 8 are really annoying.

Consider a packed bit array, with eight bits stored in each byte. And now you want to copy another bit array into it.



Since computer storage is addressed as bytes, rather than bits, you have to do a lot of bit unpacking, shifting, merging, and re-packing.

But if the offset in the destination is a multiple of eight, then you can move entire bytes across instead of having to do all the bit fiddling. This is a huge performance gain.

You're probably more used to calling these packed bit arrays "bitmaps", and bitmaps are really important in graphical environments.

The screen itself is a giant bitmap, and this means that copying data to the screen goes much faster if *x*-coordinate of the destination resides on a full byte boundary. And the most common *x*-coordinate is the left edge of a window's contents (known as its client area).

Applications can request that Windows position their windows so that their client area began at these advantageous coordinates by setting the `CS_BYTEALIGNCLIENT` style in their window class. And pretty much all applications did this because of the performance benefit it produced.

So what happened after Windows 95 that made this optimization go away?

Oh, the optimization is still there. You can still set the `CS_BYTEALIGNCLIENT` style today, and the system will honor it.

The thing that changed wasn't Windows. The thing that changed was your video card.

In the Windows 95 era, predominant graphics cards were the VGA (Video Graphics Array) and EGA (Enhanced Graphics Adapter). Older graphics cards were also supported, such as the CGA (Color Graphics Adapter) and the monochrome HGC (Hercules Graphics Card).

All of these graphics cards had something in common: They used a pixel format where multiple pixels were represented within a single byte,[1] and therefore provided an environment where byte alignment causes certain *x*-coordinates to become ineligible positions.

Once you upgraded your graphics card and set the color resolution to "256 colors" or higher, every pixel occupies at least a full byte,[2] so the requirement that the *x*-coordinate be byte-aligned is vacuously satisfied. Every coordinate is eligible.

Nowadays, all graphics cards use 32-bit color formats, and the requirement that the coordinate be aligned to a byte offset is satisfied by all *x*-coordinates.[3] The multiples of 8 are no longer special.

[1] The VGA also supported 8-bit color, but Windows didn't use it because of its extremely low resolution.

[2] The VGA so-called Mode X is an odd case because it is a planar 256-color mode, so even though every pixel occupied a full byte, you were nevertheless better off if coordinates were a multiple of 4. Windows didn't use Mode X, so that little quirk never really entered the picture.

[3] I guess you could force your video card into a 16-color mode and try to relive the world where pixels don't occupy full bytes, and some *x*-coordinates perform better than others.

Raymond Chen

Follow