Subject: Re: Time for little quiz!
Posted by mirek on Wed, 05 Apr 2006 11:14:01 GMT
View Forum Message <> Reply to Message

gprentice wrote on Wed, 05 April 2006 06:23ok, well sorry to be dumb but ... why do you want to avoid the conditional jump. Why do you think x.colorcount += (-q >> 31) & 1; is going to be more efficient than if(q) colorcount++; (or x.colorcount += bool(q);)

Most likely on any CPU since Pentium I: Failed branch prediction causes pipline stall - that is at least 12 CPU cycles wasted (31 on Prescott). Also, jumps cannot be fed to more executional units - while arithmetic ops can be. It is very likely that modern CPU will be doing those colorcount ops in the same time as operations around.

BTW, shifting 31 times is 1 cycle on AMD and PIII/Core/Conroe CPUs (unfortunately, it is more pricey for Netburst, however failed branch prediction is still even more pricey there).

Quote:
BTW - I was thinking of 64 bit machines (regarding 32 bit ints) - but compilers for 64 bit seem to have standardized on 32 bits for int still?

Yes, all major 64 bit CPUs today have 32bit ints. Well, to put it more straight, standard for all C compilers is to use 32 bit ints, other than that they support 8/16/32/64 integer quantities.

The reason is obvious - 32 bits is enough in most cases where integer quantity is to be used and 64 bit ints would eat much more space in data cache. BTW, AMD64 opcodes for 64 bit int operarations are 1 byte longer (they have "use 64 bits" prefix).

All that said, above jump-less technique would work with 64bit int as well.

Mirek