The Alpha AXP, part 14: On the strange behavior of writes to the zero register

devblogs.microsoft.com/oldnewthing/20170825-00

August 25, 2017

Raymond Chen

I noted early on that a special rule in the Alpha AXP is that an instruction that specifies the *zero* register as a destination is permitted to be completely optimized out by the processor (with two exception, as noted).

You might wonder what the point of this rule is. I mean, if you want the processor to not execute an instruction, then just don't write it!

Well, you might need to encode a nop instruction for alignment purposes, say, when falling through to the top of a hot loop, so that the loop starts at the beginning of a cache line. By giving the processor wide latitude in decide whether or not to ignore instructions which target the *zero* register, the architecture allows implementations to detect these instructions and simply remove them from the pipeline. Since it remains unspecified how many if any of the side effects of the instruction actually occur, the processor could pull it out of the pipeline at any stage of its execution, or before any of it executes at all.

The absence of a flags register means that the vast majority of instructions don't have a way to detect whether the arithmetic operation actually executed. The only one I can think of offhand is that you could use the `/V` version of an arithmetic operation to request a trap on overflow, perform an operation that intentionally overflows, and then see if a trap gets raised when you perform the `TRAPB`. But even then, that only tells you that the processor got as far as "raise an exception if an overflow was detected" step; you don't know whether it actually performed the calculation. (Though since the result is thrown away, it really doesn't matter either way.)

In practice, there's no need to detect this. In practice, you just write `BIS zero, zero, zero` for your nop instruction. If the processor optimizes it out, then great!

Exercise: What if you explicitly don't want your instruction to be optimized out? How could you express that?

Raymond Chen

Follow