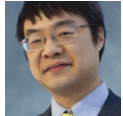


# In the red corner, EXCEPTION\_INT\_DIVIDE\_BY\_ZERO and STATUS\_INTEGER\_DIVIDE\_BY\_ZERO; and in the blue corner, EXCEPTION\_INT\_OVERFLOW and STATUS\_INTEGER\_OVERFLOW

 [devblogs.microsoft.com/oldnewthing/20141002-00](http://devblogs.microsoft.com/oldnewthing/20141002-00)

October 2, 2014



Raymond Chen

The exception code `EXCEPTION_INT_DIVIDE_BY_ZERO` (and its doppelgänger `STATUS_INTEGER_DIVIDE_BY_ZERO`) is raised, naturally enough, when the denominator of an integer division is zero. The x86 and x64 processors also raise this exception when you divide `INT_MIN` by `-1`, or more generally, when the result of a division does not fit in the destination. The division instructions for those processors take a  $2N$ -bit dividend and an  $N$ -bit divisor, and they produce  $N$ -bit quotient and remainder. Values of  $N$  can be 8, 16, and 32; the 64-bit processors also support 64. And the division can be signed or unsigned. Therefore, you can get this exception if you try to divide, say,  $2^{32}$  by 1, using a 64-bit dividend and 32-bit divisor. The quotient is  $2^{32}$ , which does not fit in a 32-bit divisor. The Windows 95 kernel does not attempt to distinguish between division overflow and division by zero. It just converts the processor exception to `EXCEPTION_INT_DIVIDE_BY_ZERO` and calls it a day. The Windows NT kernel realizes that the underlying processor exception is ambiguous and tries to figure out why the division operation failed. If the divisor is zero, then the exception is reported as `EXCEPTION_INT_DIVIDE_BY_ZERO`. If the divisor is nonzero, then the exception is reported as `EXCEPTION_INT_OVERFLOW`. Another place that `EXCEPTION_INT_OVERFLOW` can arise from a processor exception is if an application issues the `INTO` instruction and the overflow flag is set.

**Puzzle:** The `DIV` and `IDIV` instructions support a divisor in memory. What happens if the memory becomes inaccessible after the processor raises the exception but before the kernel can read the value in order to check whether it is zero? What other things could go wrong?

Raymond Chen

**Follow**

