How Bad Is the Pentium FPU Bug?

Benign for Most Users—IBM Overreacts with a Pentium Ban

by Linley Gwennap

*Since we first reported this bug (see **0816MSB.PDF**), new information has surfaced. This article incorporates data from Intel and IBM white papers on the problem as well as from other knowledgeable sources.*

As has been widely reported, Intel's Pentium processor contains a bug in the floating-point divide unit that causes inaccurate results to be generated under certain rare circumstances. This bug has led IBM to suspend sales of Pentium systems, despite Intel's claims that it is highly unlikely that the average user will ever encounter the problem. Which company is right?

We believe that, although it is possible to duplicate the problem by doing simple arithmetic in a common spreadsheet program, the chance of a normal user seeing it is extremely small. Users who perform a large number of floating-point divisions, however, are virtually guaranteed to have the problem occur at least once during the lifetime of their machines. The most worrisome feature of the bug is that, other than an incorrect result, there is no indication that a problem has occurred.

Intel discovered the problem last June through independent testing. But due to the time required to locate and fix the problem, verify the fix, and put the new version into production, the bug is present in virtually every Pentium processor shipped to date. Intel says that it will begin shipping corrected processors in volume in January and will completely switch to the new design shortly thereafter. Based on normal manufacturing and channel delays, systems with the fix will not reach customers until well into the New Year. In fact, depending on inventory levels, systems with the old CPUs may be sold to end users as late as this coming spring.

Anatomy of the Bug

The Pentium processor uses a radix-4 division algorithm, meaning that the divider generates two bits of quotient per cycle, doubling throughput over that of the 486. Pentium implements a version of the classic SRT (Sweeney, Robertson, Tocher) algorithm[1].

Simply put, on each iteration, the FPU uses the most significant bits of the divisor and the dividend (or, after the first iteration, the divisor and the remainder) to generate a "guess" of the next two bits in the quotient. This guess is taken from a lookup table. The guess is then multiplied by the divisor and subtracted from the dividend to generate a new remainder. This process is repeated until enough quotient bits are generated.

A guess can be one of five values: –2, –1, 0, 1, or 2. Using five values instead of four makes the algorithm somewhat self-correcting; if the previous guess left too large of a remainder, the quotient is adjusted in the subsequent pass using a guess of –2.

Pentium's lookup table is indexed using seven bits of the remainder and four bits of the divisor, for a total of 2,048 (2^{11}) possible entries. When this table was implemented in a PLA, five entries were inadvertently omitted. If these entries are accessed during the iterative process, an incorrect guess is generated.

The five entries correspond to divisors with the following bit patterns in the most significant bits: 1.0001, 1.0100, 1.0111, 1.1010, and 1.1101. Because of the self-correcting nature of the algorithm, however, these patterns must be followed by a string of ones to generate an incorrect result. Furthermore, since the lookup table is partially indexed by the dividend (or remainder) bits, errors will occur with some dividends but not others, even when the divisor itself contains the necessary bit pattern. Note that, because the divide hardware operates on the significands (commonly called mantissas), exponent values have no effect on the error.

Once recognized, this flaw was easily corrected by regenerating the PLA equations to cover the extra five cases. This fix added a few terms to the PLA and required a metal mask change to implement.

For uncorrected chips, all floating-point divide and remainder instructions are affected. Several transcendental instructions use the divider as well. Intel mathematicians have demonstrated that the sine, cosine, and log functions cannot trigger the problem because the limited subset of divisors they use is not susceptible. There are some cases of inaccurate results when calculating tangents, but these are extremely rare and occur only in very low-order bits.

Probability of Occurrence

Intel asserts that the fraction of all operand pairs that will generate an inaccurate result is 1.14×10^{-10}; most observers concur with this value. Thus, assuming random operands, the probability of an error is about one in nine billion. This empirical estimate correlates with more than one trillion test cases performed by Intel.

When the error occurs, the first 11 bits to the right of the binary point are always correct. The error is equally likely to cause an inaccuracy starting at any of the bits from positions 12 to 52 to the right of the binary point. These positions correspond to the fourth to fifteenth significant decimal digits. Figure 1 shows the probability of

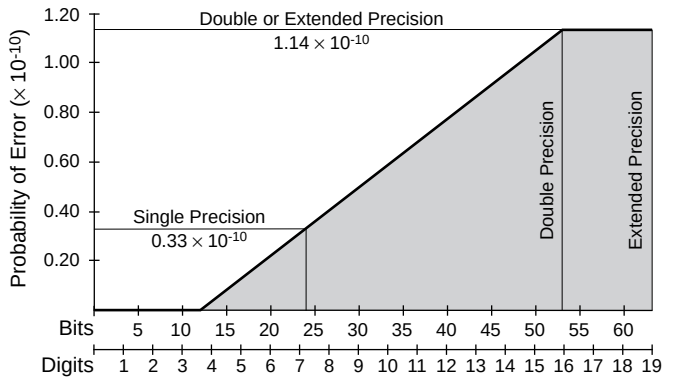

Figure 1. The probability of an error affecting a given bit position increases as the significance of the bit decreases. (Source: Intel)

an error in various binary and decimal digits.

The probability of an error affecting either the fourth or fifth significant decimal digit are 0.13×10^{-10}, or one in 100 billion. For single-precision calculations, which use 24 bits of precision, the probability of an error is 0.33×10^{-10}, or one in 30 billion. For double or extended precision, the probability is the full 1.14×10^{-10}.

Another way to characterize the problem is to look at the numbers that are affected. Table 1 lists several divisors known to generate incorrect results. Tim Coe, a floating-point designer at Vitesse, points out that the failing bit patterns correspond to the integer values 3, 9, 15, 21, and 27. These integers themselves do not cause problems because the initial bit patterns must be followed by a string of ones. Divisors very close to but slightly less than these integers, such as 2.9999995, are more likely to trigger the problem; by Coe's estimates, this type of value fails at a rate of one in 200,000, far more often than randomly selected values.

Specifically, Coe's analysis points to the following divisors as dangerous:

$3.0 > \text{divisor} \geq 3.0 - 36 \times 2^{-22}$
$9.0 > \text{divisor} \geq 9.0 - 36 \times 2^{-20}$
$15.0 > \text{divisor} \geq 15.0 - 36 \times 2^{-20}$
$21.0 > \text{divisor} \geq 21.0 - 36 \times 2^{-19}$
$27.0 > \text{divisor} \geq 27.0 - 36 \times 2^{-19}$

Divisors in these ranges are not guaranteed to fail but are statistically more likely to fail. Other potentially failing values can be generated by multiplying these values by multiples of two, since this calculation changes only the binary exponent. Coincidentally, nearly every integer multiple of three falls into the above pattern.

In normal usage, arithmetic operands are not randomly distributed. Integers, of course, are the most common and cannot directly cause an error. IBM points out that it is common to approximate a value such as 1/3 as 0.333333 (or some number of threes). If this approximation is multiplied by an integer such as 9, a dangerous value could result. IBM also notes that, due to rounding errors, certain simple calculations (e.g., 4.1 – 1.1) generate dangerous values.

By ignoring the prevalence of integers, IBM calculates the probability of generating an inaccurate quotient to be roughly one in 100 million, nearly 100 times greater than Intel's estimate. Given the near impossibility of modeling the true distribution of operands in real-world problems, we can state only that IBM's answer is likely to be an extreme upper bound on the actual probability.

Impact on Applications Varies

The impact of the bug on software depends on the type and frequency of calculations performed. Most software never uses the floating-point unit, performing all math calculations on integer values; these programs will not be affected at all. This group includes operating systems (e.g., Windows), word processors, file servers, transaction processing, and utility programs.

Many spreadsheet programs—including 1-2-3, Excel, and Quattro Pro—perform some floating-point divisions. Intel asserts that users of these programs will receive an inaccurate result once in 27,000 years, based on an average of 1,000 independent divide operations per day. (Because the bug is data-dependent, recalculating a spreadsheet will not trigger the flaw unless data has changed.)

Even when an inaccurate result is generated, Figure 1 shows that it is likely to be in a relatively insignificant decimal place. Typical financial spreadsheets that use division to calculate percentages and ratios will probably not be noticeably impacted .

Graphics applications, such as Photoshop or games with 3D graphics, make heavy use of the FPU and thus may trigger the bug. An inaccurate quotient could cause a single pixel error, but since the magnitude of the inaccuracies is so small, these artifacts will probably not be detectable by the human eye.

The impact on engineering

Mantissa (Binary Form)	Decimal Integer	Decimal (Scaled)
1011 1111 1111 1111 1111 1011 1000 0010 0011 0111 1011 0100	52,776,539,295,213	2.99999892918
1011 1111 1111 1111 1111 1011 1000 0010 0110 0100 0000 0000	206,158,356,633	2.99999892934
1011 1111 1111 1111 1111 1011 1000 0010 0111 0011 1111 0000	13,194,134,824,767	2.99999892940
1011 1111 1111 1111 1111 1011 1000 0010 0111 1110 1110 1000	26,388,269,649,885	2.99999892944
1011 1111 1111 1111 1111 1100 0000 0000 0000 0000 0000 0000	3,145,727	2.99999904633
1011 1111 1111 1111 1111 1110 1110 0000 1011 0111 0111 1100	52,776,553,426,399	2.99999973245
1011 1111 1111 1111 1111 1111 1011 1000 0010 0101 0011 0000	13,194,139,238,995	2.99999993308
1011 1111 1111 1111 1111 1111 1011 1000 0010 1001 0000 0000	824,633,702,441	2.99999993309
1000 1111 1111 1111 1110 0011 0101 1110 0001 0101 0001 0000	9,895,574,626,641	8.99997269393
1010 0111 1111 1111 1111 0110 1101 0110 0001 0010 1000 0000	1,443,107,810,341	20.99998252148

Table 1. Binary representation of several failing divisors shows a common pattern. Other failing cases begin with 11010 and 11101. The right column shows how each value can be divided by a power of two to be a number in Tim Coe's dangerous regions. (Source: Andreas Kaiser, Coe).

applications is potentially more severe. Many engineering applications—such as structural mechanics, fluid dynamics, and circuit analysis (SPICE)—perform large numbers of divide operations. These algorithms could generate up to 100 million divides per day. At this rate, an application would get about one incorrect result per year in single-precision mode or one error every three months for more precise calculations. Similar rates would apply to economists or stock traders with complex econometric models.

When the errors occur, they may be too insignificant to cause a problem. Depending on the algorithm, however, even errors in low-order bits can propagate into more significant bits as numbers are combined in new calculations. In the worst case, the result could be an incorrectly designed part or a poor financial decision.

Mathematicians with specialized problems could see errors even more frequently. The peak divide rate of a 100-MHz Pentium processor is 220 billion divide operations per day. Although it would be impossible to sustain such a rate for useful calculations, a rate of 9 billion divisions per day would cause an incorrect answer about once a day. The number of people doing this type of work on a Pentium system, however, is surely very small.

IBM Makes Unsupportable Claims

IBM's analysis uses an estimate of 5,000 divides per second during spreadsheet recalculation and claims that a spreadsheet user could spend 15 minutes per day recalculating, resulting in 4.2 million divides per day, far more than Intel's estimate. This divide rate, combined with IBM's higher probability estimate, leads to the company's claim of one failure per 24 days. IBM has used this value to support its ban on Pentium PC sales.

This reasoning, however, is transparently fallacious. IBM ignores the fact that most divides will repeat the same operations during a recalc, since only a few numbers typically change at a time. Furthermore, most spreadsheets recalculate in a fraction of a second (the visible delay is mainly screen delay); it is ludicrous that a typical user could actually be performing thousands of divides per second for 15 minutes. As a thought experiment, try to imagine a spreadsheet that, over the course of a day, contains 4.2 million *different* numbers. Finally, IBM forgets to note that more than 80% of the errors occur beyond the sixth digit, making them unlikely to be noticed by the typical spreadsheet user.

IBM's ban on Pentium sales has little technical basis. If errors were as probable as IBM claims, many more users would have noticed the problem; in fact, Intel probably would have caught the bug in prerelease testing. No other major PC vendor has adopted IBM's policy.

The move will hardly hurt Big Blue, which is a minor player in the Pentium PC market. The company is simply delaying shipment on Pentium orders until it receives corrected chips from Intel, which should take only a month or two. In fact, IBM will continue to ship Pentium systems to customers that demand them.

The ban appears to be a marketing ploy to make IBM look good and kick Intel when it's down. Although IBM is a major Intel customer, it markets x86 and PowerPC processors that compete directly with Intel's chips. Few users are likely to switch from a Pentium to an IBM processor today, but it is in IBM's long-term interest to sully both Intel and the Pentium brand name.

Little Effect on Most Users

In contrast, Intel asserts that very few Pentium users will be affected by this problem. Intel's estimate of a 27,000-year MTBF (mean time between failures) may be a bit too long, but given the smaller probability of the incorrect result being significant enough to cause a visible problem, this estimate seems reasonable.

By comparison, the MTBF of the Pentium processor itself, due to metal migration and other molecular effects, is estimated to be 200 years; the MTBF of 8M of DRAM, even with error correction, is 700 years. The MTBF of a typical Windows application is not known but is probably measured in days. Note that in most cases, however, these types of failures will cause the system to crash; the divide bug, on the other hand, gives the user no warning that an incorrect result has been generated.

For engineers and others performing complex calculations, the divide bug is worrisome, as every one of these users is likely to experience the failure sometime during the lifetime of their Pentium systems. Intel asserts, based on numerous tests, that even when errors

Testing for the Pentium Bug

All Pentium systems sold during 1994 have processors with the divide bug. Users who purchase systems in the next several months may wish to test their machine to determine whether it contains a corrected Pentium. There is no way to determine this fact visually, as the processor's heat sink covers the revision markings.

The simplest test can be performed using a common spreadsheet program, such as Microsoft Excel, by inserting the following equation into a cell:

(4195835 / 3145727) * 3145727

As is clear from visual inspection, the correct result is 4195835. This result will be achieved on a 486-based system, for example. But because this pair of numbers triggers the bug, a Pentium system will return a result of 4195579. Note that this is one of the rarest cases: an error in the fifth significant decimal digit.

The same effect can be achieved using a program like Microsoft Windows Desk Calculator. Software types can code the above sequence into a short C routine to test the processor (an exercise left to the reader).

For More Information

If you believe that you qualify for Intel's Pentium replacement program or wish to obtain a copy of the company's 31-page analysis of the FPU bug, contact Intel (Santa Clara, Calif.) at 800.628.8686 or 916.356.3551, or contact your system vendor.

occur, they are not likely to cause a "meaningful inaccuracy" in engineering calculations. The company places the odds of such a meaningful failure at one per 1,000 years. We recommend that these users, to be safe, seek a corrected Pentium for future work.

For mathematicians and a few other researchers, the odds of a failure are far too high to be acceptable. These users must perform future calculations on a system without this flaw, using either a corrected Pentium or a non-Pentium processor. Furthermore, any such calculations performed on a Pentium system to date must be considered suspect and probably need to be redone.

Intel Responds with Replacement Program

We estimate that 4–5 million Pentiums have been shipped with the bad divider. For Intel to do a complete recall of these chips would be impractical. Unlike software, a Pentium processor is not user-upgradable; Pentium system owners have to take their machines to a service center to be repaired. Given an average manufacturing cost of $150 per Pentium, plus an allowance of $100 for shipping and service costs, it would cost Intel more than $1 billion to perform a total recall.

Instead, the company has established a hot line (see information box) for end users to request replacement parts. Callers who claim to be doing heavy floating-point work will be sent a new, corrected Pentium chip. A $400 deposit (credit cards accepted) is required to ensure that the original Pentium is returned, foiling seekers of free processors. The company claims a two-week turnaround for replacement parts and says that it has already shipped more than a thousand such devices.

For callers using typical business or consumer software, Intel attempts to dissuade them from seeking a replacement part. It points out, for example, that opening the system and replacing the CPU is more likely than the divide bug to cause a system failure. The company says that it convinces most casual users they don't need a replacement but that it will ultimately give a new Pentium to any who demands one.

Most major PC makers have set up their own replacement programs. Users who bought Pentium PCs from reputable sources should contact their vendor directly to have their systems upgraded.

It is also possible to work around this bug in software. Lotus, for one, has released a version of 1-2-3 that functions properly even on an original Pentium. Intel is developing a compiler patch that will allow any software developer to do the same, simply by recompiling its application. These software fixes check each divisor and, if it is dangerous, adjust both divisor and dividend by 7/8 to avoid the bug. Because the check is quick and most divisors pass, the performance impact is less than 5%. A software fix, however, will help only those users who acquire new software.

A Public Relations Nightmare

The Pentium FPU bug touched off a flurry of criticism on the Internet and other electronic bulletin boards. The attention spilled into both the industry press and the popular press, where the response has been more varied. While some press accounts have been well reasoned, others have been near-hysterical. Intel's stock price has dropped by about 5% since the bug become public but continues to be buoyed by heavy buying each time the price dips on bad news.

Intel's handling of this problem has not been particularly enlightened *(see **0817ED.PDF**)*, but it is unlikely to have a significant financial impact on the company. Pentium PC sales have continued strongly even as the bug has been widely publicized. Some businesses have delayed but not canceled purchases. Most corporate buyers realize that, while Pentium is not perfect, there are few viable alternatives, none of which is likely to be less bug-ridden than the Intel chip. The cost of the replacement program will probably be less than $10 million, about what Intel spends on advertising each month.

One cause for concern is the possibility of legal damages. Intel could be considered negligent for not disclosing the bug promptly. To win a legal case, however, a plaintiff would have to demonstrate that an incorrect result caused significant financial loss, a difficult task. Now that Intel has notified users of the problem, it can probably escape further liability. Several lawsuits have been filed so far, and Intel is rumored to have issued at least one six-figure settlement.

The biggest long-term impact may be among members of the scientific and engineering community. Many of these users have avoided Intel processors for years because the 386 and 486 lacked the math performance needed in these markets. Intel specifically concentrated on boosting the floating-point performance of Pentium to improve its standing among technical users and, until recently, had been gaining momentum. The lingering effect of this bug will be to blunt Intel's thrust into the technical market. ♦

[1] For a detailed description of SRT division, see "Higher-Radix Division Using Estimates of the Divisor and Partial Remainders," Daniel E. Atkins, *IEEE Transactions on Computing*, C 17:925–935 (1968).

