

The “Pentium Bug” and SRT Division P415/P515, Spring 2004

- Chronology
- Humor
- The Questions
- SRT Division
 - Illustration
 - Development of the Algorithm
- The Bug
- Discussion
 - What caused it?
 - Why wasn’t it found?
 - Impact
 - Lessons

The “Pentium Bug” Chronology [Bryant 98]

1993

March:

◇ Intel introduces the Pentium^R

1994

June:

◇ Prof. Thomas Nicely, Lynchburg College, reports errors in calculating twin primes reciprocals.

October:

◇ After considerable background discussion, word starts circulating on the Internet.

◇ Others confirm error and find more instances.

November:

◇ Tim Coe, of Vitess Semiconductor, proposes a [substantially correct] software model explain the cause.

◇ An Intel internal report analyzes a flaw in the Pentium FDIV instruction.

◇ Intel CEO Andy Grove responds (Nov. 24, 1994):

- Minor bug known at Intel since mid-94.
- All micros have bugs.
- “Average user” will never see the problem (MTBE: 27,000 years).
- Most applications don’t fewer than 1,000 division a day.
- FDIV error rate is about 1.5×10^{-9}
- Error conditions guarantee small errors.
- Many applications (e.g. graphics) can tolerate occasional small errors.
- Offers replacement for justified need.

◇ Popular press generally accepts Intel’s claims about “obscure error.”

◇ Intel confirms 2 million defective chips have been shipped.

December 1994:

◇ IBM disagrees (MTBE: 24 days); stops shipment of Pentium based PCs.

- Even casual spreadsheet users may do about 4.2×10^6 divides per day.
- The error distribution is not uniform.
- Under some reasonable conditions FDIV error rate can approach 10^{-2}

◇ Some question IBM's motives.

◇ A flurry of Internet communication condemns Intel's attitude and questions its evaluation of the problem.

◇ Intel revises replacement policy. Hard to interpret policy but easy to accomplish in practice. 2% of home users and 10% businesses eventually get replacements.

◇ Intel (Andy Grove) admits it mishandling the problem, but stands by its evaluation.

◇ Public perception is that Intel was responsive \Rightarrow positive publicity (\$? worth?).

1995

March:

◇ Coe, *et al.* article appears in *IEEE Journ. Computational Sci. and Eng.*

May:

◇ Lamport article appears at *TAPSOFT*.

◇ Kahan posts should-have-known SRT test article.

1996

◇ Intel establishes the world's largest verification division, dominating industrial research through 2002.

◇ Reported cost of the Pentium affair reportedly \$450 million; \$15/\$16 billion market in 1996.

1997–2000

◇ All major μ processor manufacturers adopt formal verification.

◇ Surge in CAD industry tool offerings.

◇ Significant research results appear in floating point verification.

2000–2002

◇ Articles, conference panel sessions on verification “culture.”

◇ IC technology roadmap: looming “design crisis.”

1 Pentium Humor, circa November 1994

Q: How many Pentium designers does it take to screw in a light bulb?

A: 1.99904274017, but that's close enough for non-technical people.

* * *

Q: What do you get when you cross a Pentium PC with a research grant?

A: A mad scientist.

* * *

Q: What's another name for the "Intel Inside" sticker they put on Pentiums?

A: The warning label.

* * *

Q: What do you call a series of FDIV instructions on a Pentium?

A: Successive approximations.

* * *

Q: Complete the following word analogy: Add is to Subtract as Multiply is to: 1) Divide 2) ROUND 3) RANDOM 4) On a Pentium, all of the above

A: Number 4.

* * *

Q: What algorithm did Intel use in the Pentium's floating point divider?

A: "Life is like a box of chocolates." (Source: F. Gump of Intel)

* * *

Q: Why didn't Intel call the Pentium the 586?

A: Because they added 486 and 100 on the first Pentium and got 585.999983605.

* * *

Q: According to Intel, the Pentium conforms to the IEEE standards 754 and 854 for floating point arithmetic. If you fly in aircraft designed using a Pentium, what is the correct pronunciation of "IEEE"?

A: Aaaaaaaiiiiiiiiiieeeeeeeeeeee!

Top Ten New Intel Slogans for the Pentium^R

- 9.9999973251 It's a FLAW, Dammit, not a Bug
- 8.9999163362 It's Close Enough, We Say So
- 7.9999414610 Nearly 300 Correct Opcodes
- 6.9999831538 You Don't Need to Know What's Inside
- 5.9999835137 Redefining the PC – and Mathematics As Well
- 4.9999999021 We Fixed It, Really
- 3.9998245917 Division Considered Harmful
- 2.9991523619 Why Do You Think They Call It *Floating* Point?
- 1.9999103517 We're Looking for a Few Good Flaws
- 0.9999999998 The Errata Inside

For a sequence, $\{q_i\}_{[n]}$ of digits, let $\llbracket Q \rrbracket_m$ denote

$$\sum_{k=0}^{m-1} \frac{q_k}{10^k}$$

Algorithm for *long division*. Dividend P and divisor D have been normalized to be less than 10. The partial values $\{p_i\}_{[n]}$ are for discussion purposes only.

```

{1 ≤ P, D < 10}
p0, d, i := P, D, 0;
while i < n do {INV1 ∧ INV2}
  b
  choose qi ∈ {0...9} such that...
  {pi - qi · d < d}
  pi+1, i = 10(pi - qi · d), i + 1
  e
  {P - ⌊Q⌋ · d <  $\frac{d}{10^n}$ }

```

Hence, on termination

$$P/D - \llbracket Q \rrbracket_n \cdot D < \frac{D}{10^{-n}}$$

“The remainder is ‘smaller than’ D ”

Since

$$\begin{aligned} p_3 &= 10(p_2 - q_2d) \\ &= 10(10(p_1 - q_1d) - q_2d) \\ &= 10(10(10(p_0 - q_0d) - q_1d) - q_2d) \\ &= 10^3p_0 - 10^3q_0d - 10^2q_1d - 10^1q_2d \\ &= 10^3P - \left(\sum_{k=0}^2 10^{3-k} q_k d \right) \cdot d \\ &= 10^3P - \llbracket Q \rrbracket_3 \cdot d \end{aligned}$$

The loop invariants are,

$$\text{INV1} \equiv p_i = 10^i P - \llbracket Q \rrbracket_i \cdot d$$

$$\text{INV2} \equiv p_i < 10^i \cdot D$$

Independent of the radix (base). Pentium I radix is 4.

LONG DIVISION

$$\begin{array}{r}
 00402R18 \\
 \hline
 53 \) \ 21324 \\
 \underline{0 \ - \ - \ - \ -} \\
 21 \ - \ - \ - \\
 - \ 0 \ - \ - \ - \\
 \hline
 213 \ - \ - \\
 - \ 212 \ - \ - \\
 \hline
 12 \ - \\
 - \ 0 \ - \ - \\
 \hline
 124 \\
 - \ 106 \\
 \hline
 18
 \end{array}$$

"RESTORING" DIVISION

$$\begin{array}{r}
 10 \\
 00302R18 \\
 \hline
 53 \) \ 21324 \\
 \underline{0 \ - \ - \ - \ -} \\
 21 \ - \ - \ - \\
 - \ 0 \ - \ - \ - \\
 \hline
 213 \ - \ - \\
 - \ 159 \ - \ - \\
 \hline
 542 \ - \\
 - \ 530 \ - \\
 \hline
 124 \\
 - \ 106 \\
 \hline
 18
 \end{array}$$

$$\begin{array}{r}
- 0 0 0 9 7 \\
+ 0 0 5 0 0 \text{ R } -35 \quad \{500 - 97 \text{ R}-35 \\
\hline
= 403 \text{ R}-35 \\
53) 2 1 3 2 4 \quad = 402 \text{ R}+18\} \\
\quad 2 1 3 \\
- 2 6 5 \\
\hline
- 5 1 8 \quad = -520 + 2 \\
- - 4 7 7 \\
\hline
- \quad 4 0 6 \quad = -410 + 4 \\
- - \quad 3 7 1 \\
\hline
- \quad \quad 3 5
\end{array}$$

Restoring division

Maintaining positive and negative quotients, we can *sometimes* recover from an invalid guess for the next digit.

We do not have to backtrack if the guess is off a little bit.

This refinement is *not* “error correcting,” if the guess is too far off.

Digit selection

In radix 4, the choice of quotient digits is $\{-3, -2, -1, 0, 1, 2, 3\}$. Restricting this to $\{-2, -1, 0, 1, 2\}$ reduces the partial multiplications to shifts.

Keeping the guess close enough.

$$2 + \frac{2}{4} + \frac{2}{8} + \cdots + \frac{2}{4^i} + \cdots = \frac{8}{3}$$

This follows from

$$a + ar + ar^2 + \cdots + ar^n = \frac{a(i - r^{n+1})}{1 - r}$$

We need to keep p_{k+1} within the range $[-\frac{8}{3}d, \frac{8}{3}d]$

The division algorithm is now

```

{1 ≤ P, D < 2}
p0, d, i := P, D, 0;
while i < n do {INV1 ∧ INV2}
  b
  choose qi ∈ {−2...+2} such that...
  {|4(pi − qi · d)| ≤  $\frac{8}{3}d$ }
  pi+1, i = 4(pi − qi · d), i + 1
  e
  {P − [Q] · d <  $\frac{d}{4^n}$ }

```

where

$$[Q]_m = \sum_{k=0}^{m-1} \frac{q_k}{4^k}$$

$$\text{INV1} \equiv p_i = 4^i P - [Q] \cdot d$$

$$\text{INV2} \equiv p_i < 4^i \cdot D$$