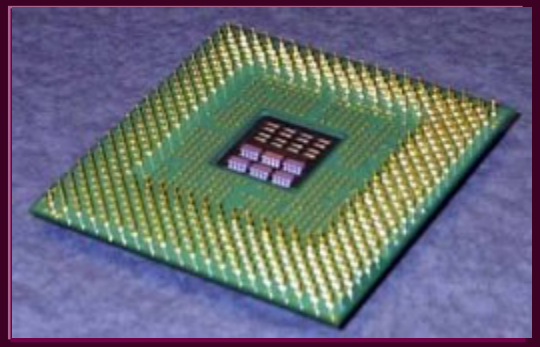


The Intel Pentium Division Flaw

Alan Edelman

Department of Mathematics

Laboratory for Computer Science

Massachusetts Institute of Technology

MIT

Not So Well Known

The bug itself is (mathematically) neat!

A Lesson (for me anyway)

So much incomplete information is out there.

Interesting Related Topics (but my topic is the bug)

- Risk to Pentium owners
- Intel's chip replacement blunder
- Kahan's SRT division tester
- Moler, Coe, and Mathisen software workaround
- Only the lawyers get rich
- Those ubiquitous Pentium jokes

Outline

- Nicely's Discovery
- Computer Science Prerequisites
- Division (SRT=Sweeney,Robertson,Tocher)
- Pentium Lookup Table
- Division Example
- Six Ones Result
  - Inequality Analysis
  - "Send More Money" Puzzle
- Always nearly five good digits

Nicely's Twin Prime Bug Discovery

- Twin primes: (5,7) (11,13) (17,19) (29,31) ...
- Nicely was summing twin prime reciprocals:
- S = 1/5 + 1/ 7 + 1/11 + 1/13 + 1/17 + 1/19 + ...
- S is finite.
- Nicely computed on many platforms.
- Nicely checked his work.

Computer Science Prerequisites

- Carry Save Addition.
- One's vs. Two's Complement.

Carry-save Addition

```
  12
  21
+ 19
----
  52
```

Carry-save Addition

```
  12
  21
+ 19
----
  52

  20
```

Answer (mod 32)

Carry-save Addition

```
  12      01100
  21      10101
+ 19    + 10011
----    -------
  52      10100

  20
```

Answer (mod 32)

Carry-save Addition

```
  12      01100      01100
  21      10101    + 10101
+ 19    + 10011    ------
------  --------       1   Sum Bits (s)
  52      10100        0   Carry Bits (c)

  20
```

Answer (mod 32)

Carry-save Addition

```
  12      01100      01100
  21      10101    + 10101
+ 19    + 10011    ------
------  ------         1   Sum Bits (s)
  52      10100        0   Carry Bits (c)
  20
```

Answer (mod 32)

$s = x \oplus y \oplus z = x+y+z \pmod 2$

$c = x \wedge y \vee x \wedge z \vee y \wedge z = (x+y+z \mid 2)$

Carry-save Addition

```
  12      01100      01100
  21      10101    + 10101
+ 19    + 10011    ------
------  --------       01   Sum Bits (s)
  52      10100        00   Carry Bits (c)

  20
```

Answer (mod 32)

$s = x \oplus y \oplus z = x+y+z \pmod 2$

$c = x \wedge y \vee x \wedge z \vee y \wedge z = (x+y+z \mid 2)$

Carry-save Addition

```
  12      01100      01100
  21      10101    + 10101
+ 19    + 10011      11001   Sum Bits (s)
  52      10100      01000   Carry Bits (c)
  20
```

Answer (mod 32)

$s = x \oplus y \oplus z = x+y+z \pmod 2$

$c = x \wedge y \vee x \wedge z \vee y \wedge z = (x+y+z \mid 2)$

Carry-save Addition

```
  12      01100      01100
  21      10101    + 10101
+ 19    + 10011      11001   Sum Bits (s)
  52      10100      01000   Carry Bits (c)
  20                +10011
```

Answer (mod 32)

$s = x \oplus y \oplus z = x+y+z \pmod 2$

$c = x \wedge y \vee x \wedge z \vee y \wedge z = (x+y+z \mid 2)$

Carry-save Addition

```
  12     01100      01100
  21     10101    + 10101
+ 19   + 10011      11001   Sum Bits (s)
  52     10100      01000   Carry Bits (c)
                   + 10011
  20                 ↑
                   00010   Sum Bits (s)
                   10010   Carry Bits (c)
```

Answer (mod 32)

$s = x \oplus y \oplus z = x+y+z \pmod 2$

$c = x \wedge y \vee x \wedge z \vee y \wedge z = (x+y+z \mid 2)$

One's vs. Two's Complement

	Two's Complement
3	00011
2	00010
1	00001
0	00000
-1	11111
-2	11110
-3	11101

One's vs. Two's Complement

	Two's Complement	One's Complement
3	00011	00011
2	00010	00010
1	00001	00001
0	00000	00000
-1	11111	11110
-2	11110	11101
-3	11101	11100

One's vs. Two's Complement

	Two's Complement	One's Complement
3	00011	00011
2	00010	00010
1	00001	00001
0	00000	00000
-1	11111	11110
-2	11110	11101
-3	11101	11100

- Division Algorithms:

Long Division Example

```
  1.42857
7|10.00000
  7
  30
  28
   20
   14
    60
```

Long Division Example

q_0 $q_1 q_2 q_3 q_4 q_5$ ← Chosen to satisfy usual inequalities

```
  1.42857
7|10.00000
  7
  ---
  30
  28
  --
   20
   14
   --
    60
```

Long Division Example

$p = 10$, $d = 7$

$10d = 70$

$p_{k+1} = 10(p_k - q_k d)$

$0 \le p_{k+1} < 70$

$q_0 \quad q_1 q_2 q_3 q_4 q_5$

```
  1.42857
7|10.00000   p0
  7
  --
  30         p1
  28
  --
   20        p2
   14
   --
    60       p3
```

Long Division Radix 10

Compute $q = p / d$.

$p_0 := p$

for k=0,1,...
    Find the digit $q_k \in \{0, 1, 2, \ldots, 9\}$ such that
    $p_{k+1} := 10(p_k - q_k d)$ satisfies $p_{k+1} \in [0, 10)d$
end

$$q = p / d = \sum_{i=0}^{\infty} q_i / 10^i$$

SRT Division Radix 4

Compute $q = p / d$.

$p_0 := p$

for k=0,1,...
    Look up a digit $q_k \in \{-2,-1,0,1,2\}$ such that
    $p_{k+1} := 4(p_k - q_k d)$ satisfies $|p_{k+1}| \leq (8/3)d$
end

$$q = p / d = \sum_{i=0}^{\infty} q_i / 4^i$$

Such q_k exists?

Algorithm correct?

Such q_k exists?

Given $1 \le p,d < 2$. Compute $q = p/d$.

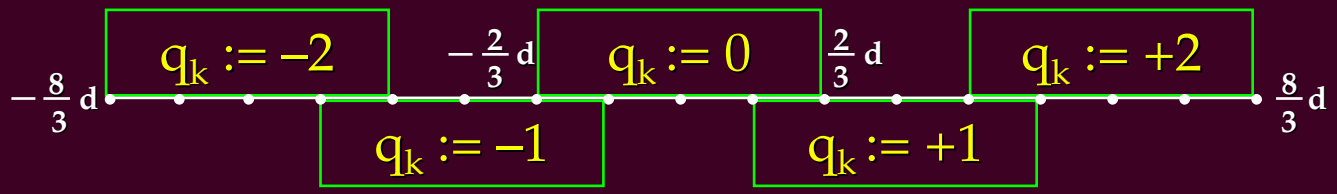
$p_0 := p$

for k=0,1,...

Look up a digit $q_k \in \{-2,-1,0,1,2\}$ such that $p_{k+1} := 4(p_k - q_k d)$ satisfies $|p_{k+1}| \le (8/3)d$

end

$$q = p/d = \sum_{i=0}^{\infty} q_i / 4^i$$



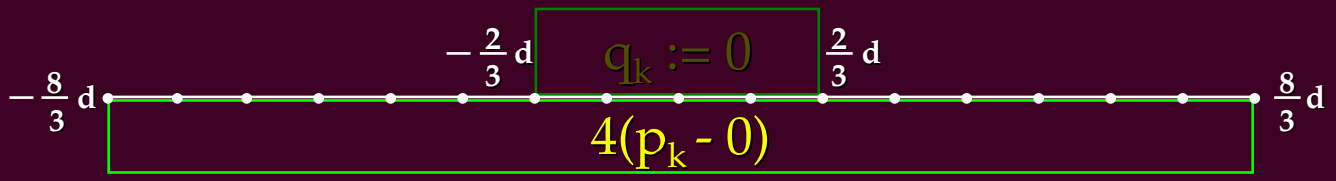
Such q_k exists?

Given $1 \le p,d < 2$. Compute $q = p/d$.

$p_0 := p$

for k=0,1,...
  Look up a digit $q_k \in \{-2,-1,0,1,2\}$ such that
  $p_{k+1} := 4(p_k - q_k d)$ satisfies $|p_{k+1}| \le (8/3)d$
end

$$q = p/d = \sum_{i=0}^{\infty} q_i / 4^i$$



$$-2/3d \le p_k \le 2/3d$$

$$p_{k+1} := 4(p_k - 0)$$

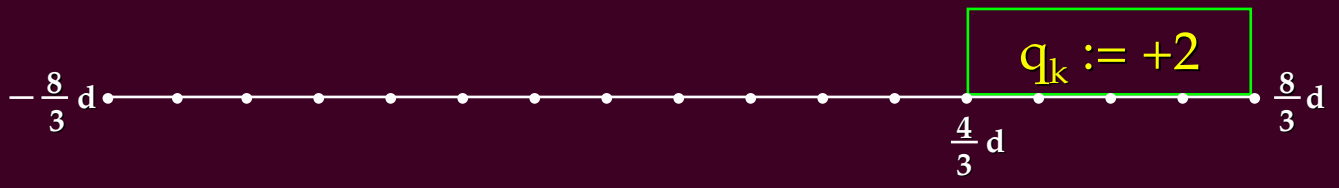
Such q_k exists?

Given $1 \le p,d < 2$. Compute $q = p/d$.

$p_0 := p$

for k=0,1,...
  Look up a digit $q_k \in \{-2,-1,0,1,2\}$ such that
  $p_{k+1} := 4(p_k - q_k d)$ satisfies $|p_{k+1}| \le (8/3)d$
end

$$q = p/d = \sum_{i=0}^{\infty} q_i / 4^i$$



$$4 / 3d \le p_k \le 8 / 3d$$

$$p_{k+1} := 4(p_k - 2d)$$

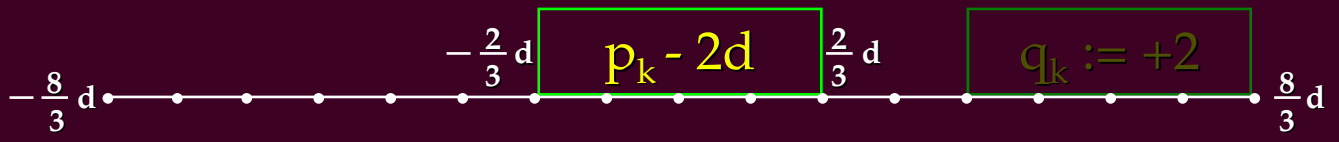
Such q_k exists?

Given $1 \le p,d < 2$. Compute $q = p/d$.

$p_0 := p$

for k=0,1,...
  Look up a digit $q_k \in \{-2,-1,0,1,2\}$ such that
  $p_{k+1} := 4(p_k - q_k d)$ satisfies $|p_{k+1}| \le (8/3)d$
end

$$q = p/d = \sum_{i=0}^{\infty} q_i / 4^i$$



$4/3 \le p_k \le 8/3$

$p_{k+1} := 4(p_k - 2d)$

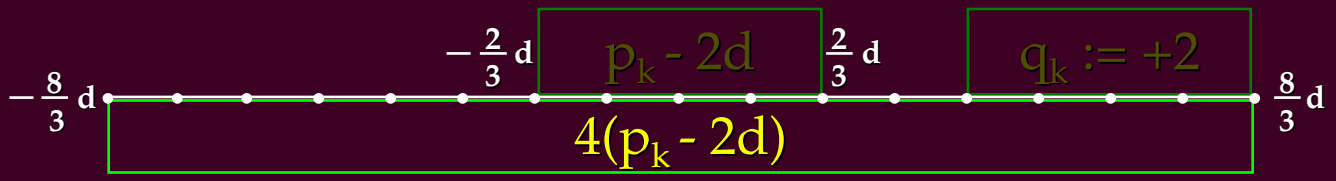
Such q_k exists?

Given $1 \le p,d < 2$. Compute $q = p/d$.

$p_0 := p$

for k=0,1,...
  Look up a digit $q_k \in \{-2,-1,0,1,2\}$ such that
  $p_{k+1} := 4(p_k - q_k d)$ satisfies $|p_{k+1}| \le (8/3)d$
end

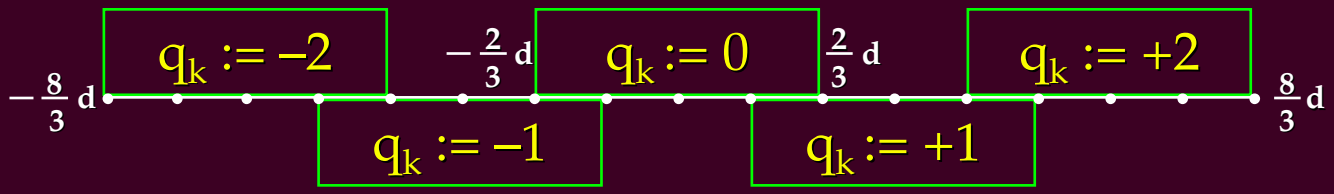
$$q = p/d = \sum_{i=0}^{\infty} q_i / 4^i$$



$$4 / 3d \le p_k \le 8 / 3d$$

$$p_{k+1} := 4(p_k - 2d)$$

A q_k For Every Point



Algorithm Correct?

Claim:

$$\frac{p}{d} = q_0 + \frac{q_1}{4} + \ldots + \frac{q_{k-1}}{4^{k-1}} + \frac{p_k}{d} 4^{-k}$$

Proof by Induction:

$$p_{k+1} = 4(p_k - q_k d) \Rightarrow$$

$$\frac{p_k}{d} 4^{-k} = \frac{q_k}{4^k} + \frac{p_{k+1}}{d} 4^{-(k+1)}$$

Algorithm Correct?

Claim:

$$\frac{p}{d} = q_0 + \frac{q_1}{4} + \ldots + \frac{q_{k-1}}{4^{k-1}} + \frac{q_k}{4^k} + \frac{p_{k+1}}{d} 4^{-(k+1)}$$

Proof by Induction:

$p_{k+1} = 4(p_k - q_k d) \Rightarrow$

$$\frac{p_k}{d} 4^{-k} = \frac{q_k}{4^k} + \frac{p_{k+1}}{d} 4^{-(k+1)}$$

Algorithm Correct?

Claim:

$$\frac{p}{d} = q_0 + \frac{q_1}{4} + \ldots + \frac{q_{k-1}}{4^{k-1}} + \frac{q_k}{4^k} + \frac{p_{k+1}}{d} 4^{-(k+1)}$$

Proof by Induction:

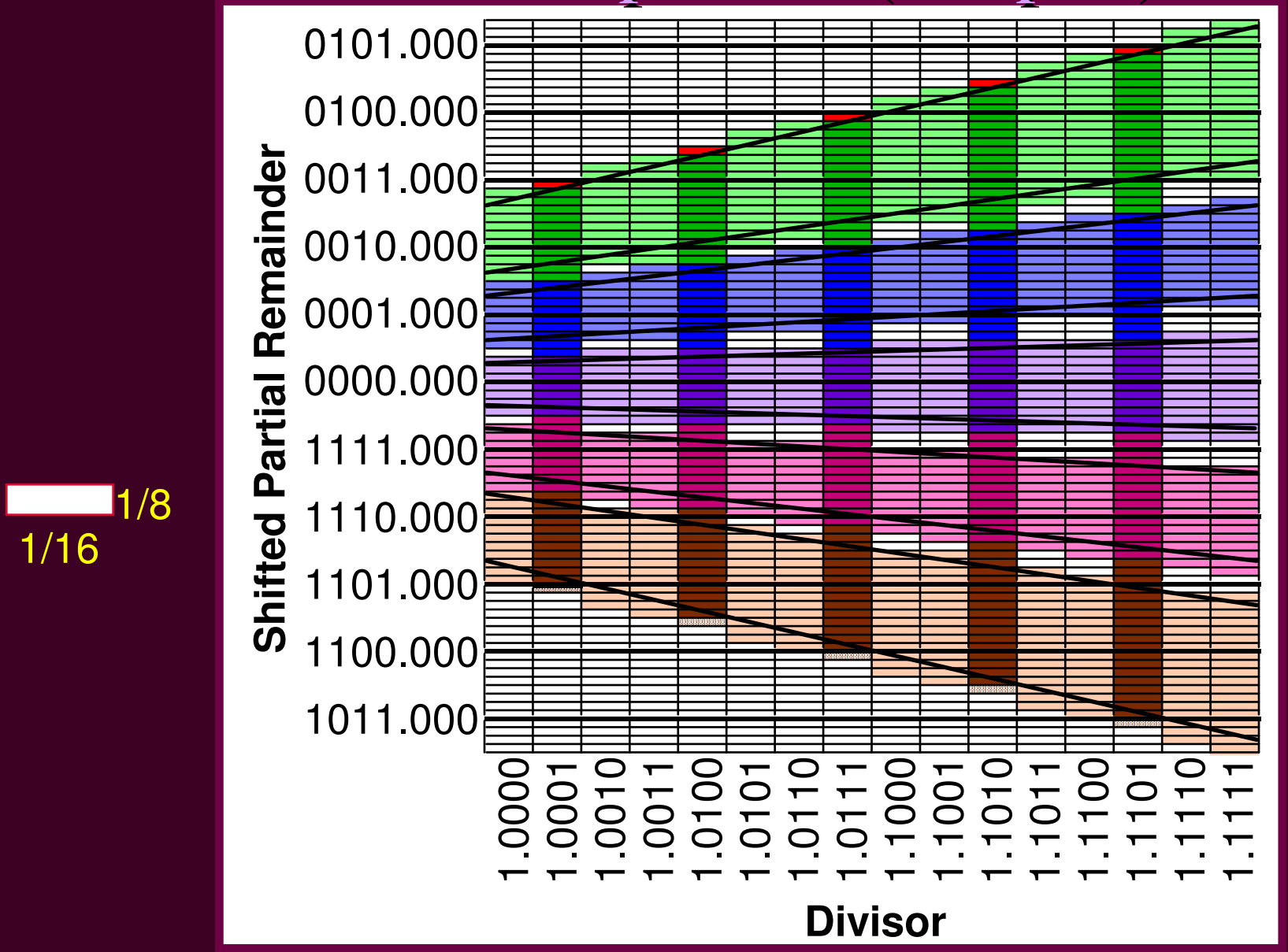
$$p_{k+1} = 4(p_k - q_k d) \Rightarrow$$

$$\frac{p_k}{d} 4^{-k} = \frac{q_k}{4^k} + \frac{p_{k+1}}{d} 4^{-(k+1)}$$

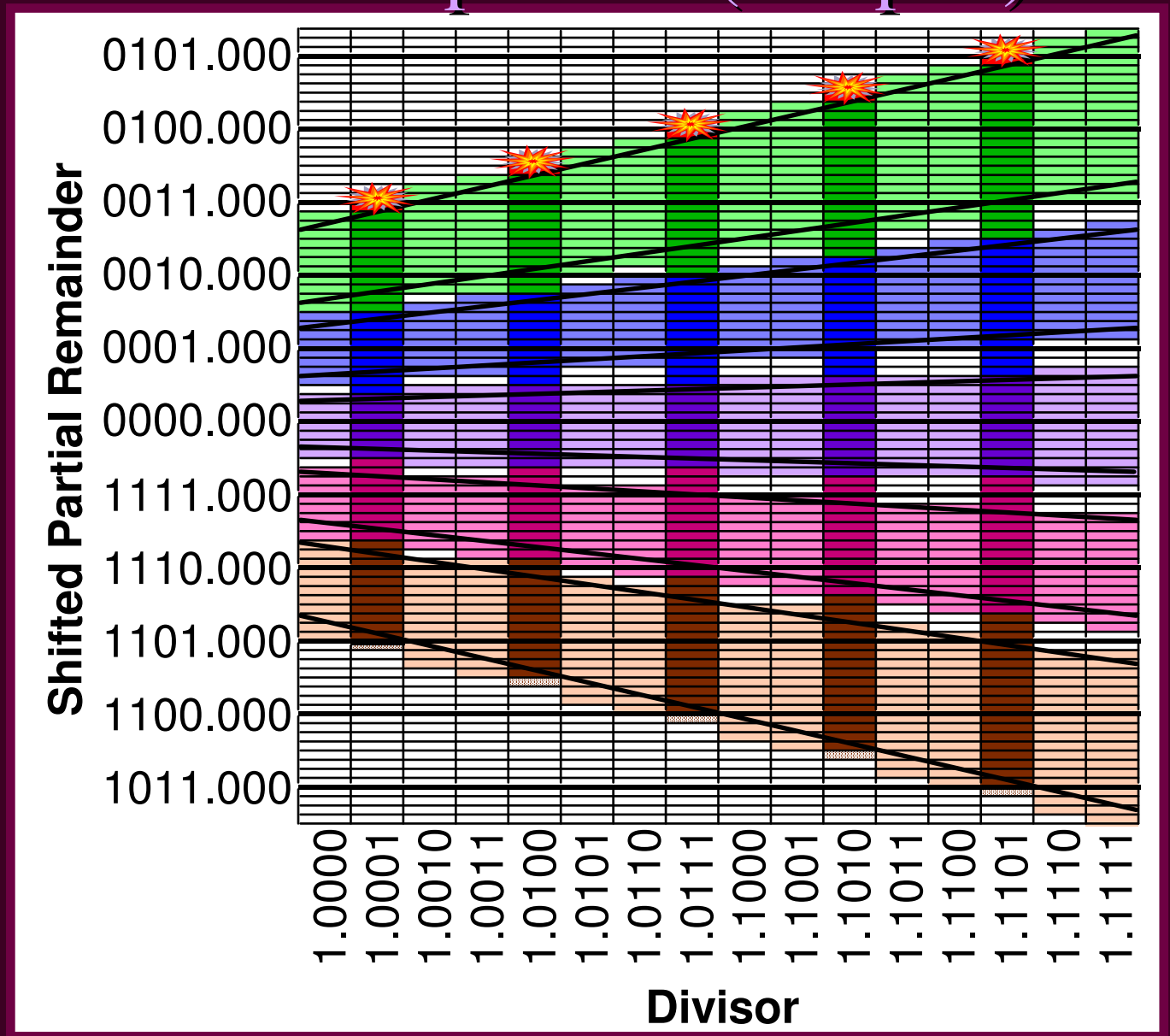
Letting $k \to \infty$ proves

$$\frac{p}{d} = q_0 + \frac{q_1}{4} + \frac{q_2}{4^2} + \ldots$$

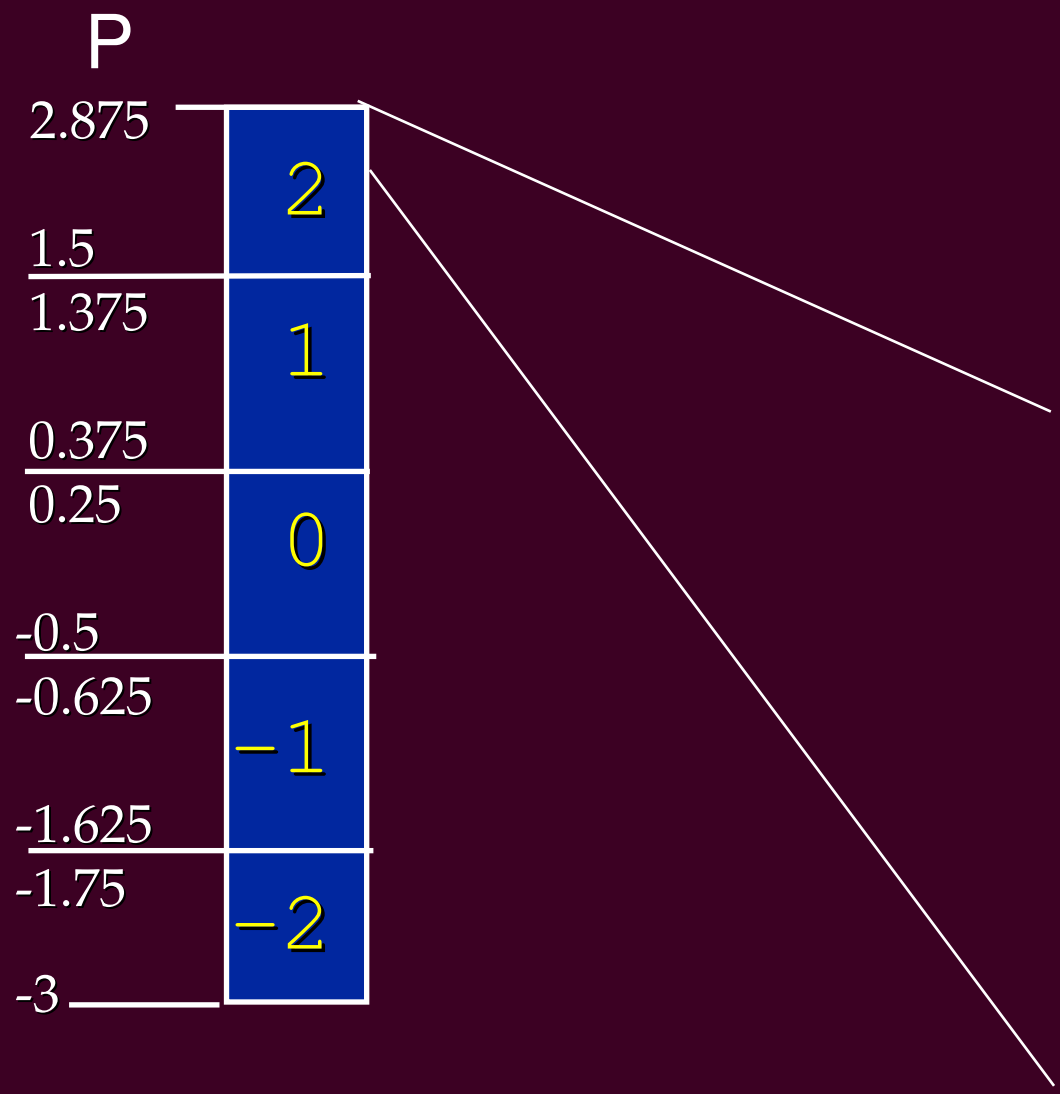
Pentium Lookup Table (P-d plot)



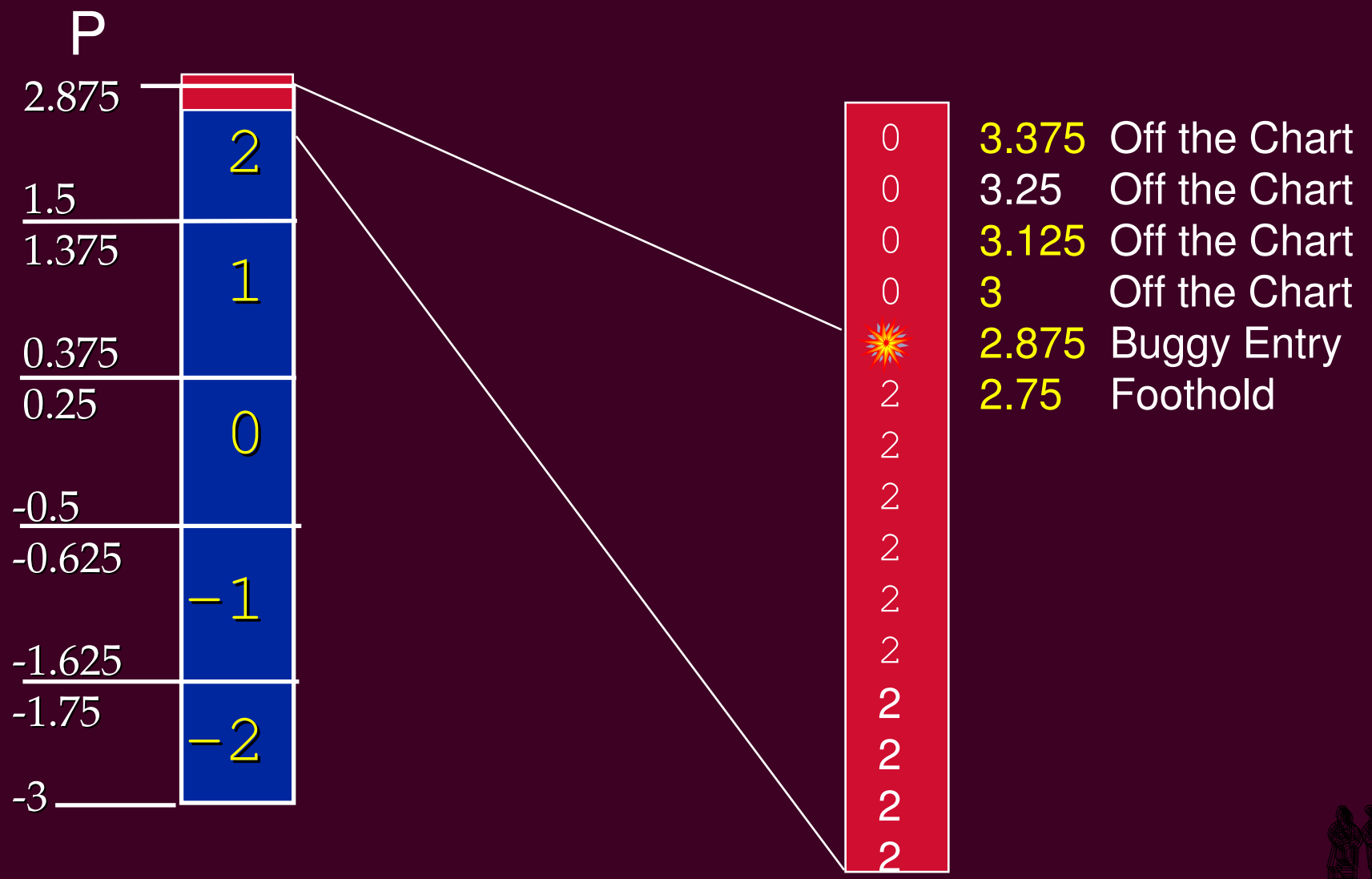
Pentium Lookup Table (P-d plot)



A Close-up Look at One Column (D=1.0001)

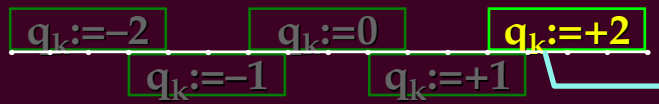


A Close-up Look at One Column (D=1.0001)



Pentium Division Example: 1.875/1.000

1.875 = { `0001.111` `00000000000` S
`0000.000` `00000000000` C



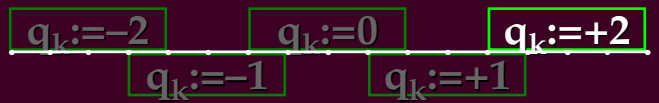
$p_{k+1} := 4(p_k - q_k d)$

Pentium Division Example: 1.875/1.000

$1.875 = \{$ `0001.111 00000000000` S
`0000.000 00000000000` C

$-2 \times 1 =$ `1101.111 11111111111`

$-0.125 = \{$ `1100.000 11111111111` S
`0011.110 00000000001` C

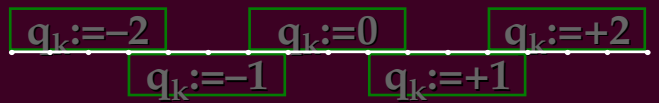


$$p_{k+1} := 4(p_k - q_k d)$$

Pentium Division Example: 1.875/1.000

$1.875 = \{$ `0001.111 00000000000` S
`0000.000 00000000000` C

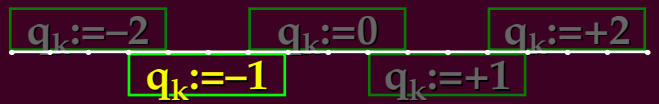
$-2 \times 1 =$ `1101.111 11111111111`



$-0.125 \times 4 = -0.5 = \{$ `0000.011 11111111100` S
`1111.000 00000000100` C

$p_{k+1} := 4(p_k - q_k d)$

Pentium Division Example: 1.875/1.000



$$1.875 = \begin{cases} 0001.111\ 00000000000\ \text{S} \\ 0000.000\ 00000000000\ \text{C} \end{cases}$$

$$-2 \times 1 = \underline{1101.111\ 11111111111}$$

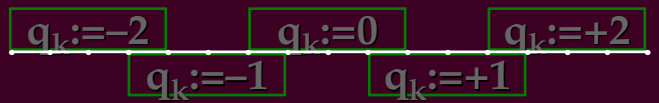
$$\begin{array}{l} 0000.011\ 11111111100\ \text{S} \\ 1111.000\ 00000000100\ \text{C} \end{array}$$

$$-(-1) \times 1 = \underline{0001.000\ 00000000000}$$

$$\begin{array}{l} 1001.111\ 11111100000\ \text{S} \\ 1000.000\ 00000100000\ \text{C} \end{array}$$

$$p_{k+1} := 4(p_k - q_k d)$$

Pentium Division Example: 1.875/1.000



$1.875 =$ {	0001.111 00000000000	S
	0000.000 00000000000	C
$-2\times1 =$	1101.111 11111111111	
	0000.011 11111111100	S
	1111.000 00000000100	C
$--1\times1 =$	0001.000 00000000000	
	1001.111 11111100000	S
	1000.000 00000100000	C
$-2\times1 =$	1101.111 11111111111	
	0000.000 00011111100	S
	1111.111 11100000100	C
$-0\times1 =$	0000.000 00000000000	
	1111.111 11111100000	S
	0000.000 00000100000	C

$2/1 + -1/4 + 2/16 + 0/64 = 1.875/1.000$

Inequality Analysis

$$P_k \le p_k \le P_k + 1/4$$

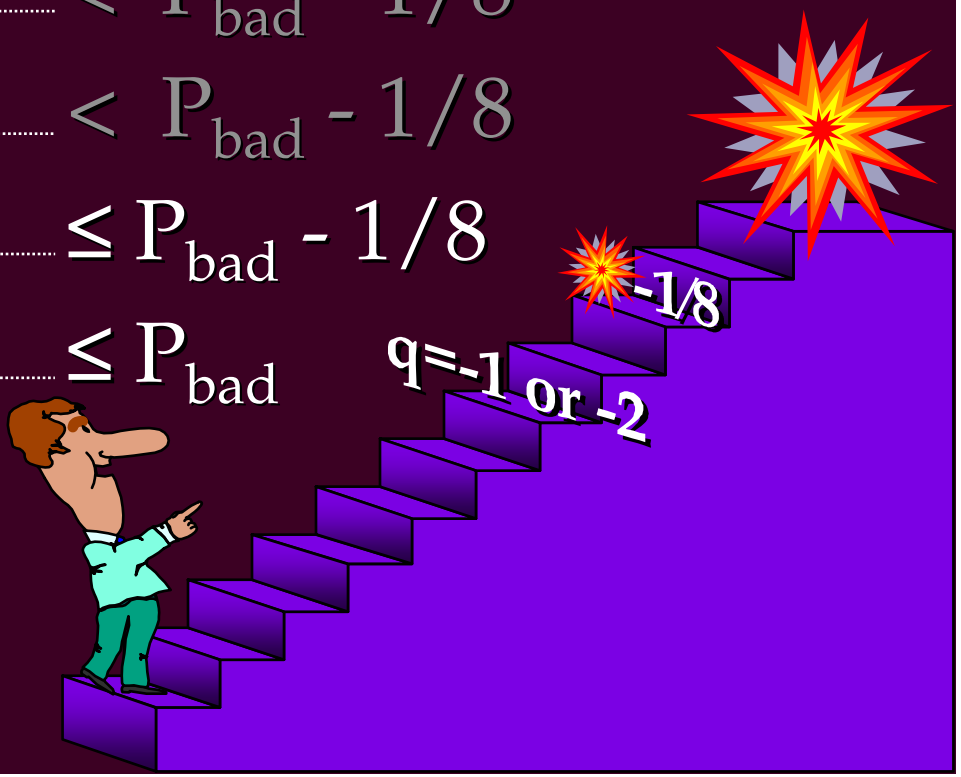
$$D \le d \le D_+ \equiv D + 1/16$$

$$P_{k+1} = 4(P_k - q_k D_+) + R_k$$

$$R_k \le R_k^{Max} \equiv \begin{cases} 3/4 & \text{if } q_k = -2 \\ 3/4 & \text{if } q_k = -1 \\ 3/4 & \text{if } q_k = 0 \\ 1 & \text{if } q_k = 1 \\ 5/4 & \text{if } q_k = 2 \end{cases}$$

Reaching the Flaw is Not Easy!

q_k	P_{k+1}
–2	$\leq P_{bad} - 1/8$
–1	$\leq P_{bad} - 1/8$
0	$< P_{bad} - 1/8$
1	$< P_{bad} - 1/8$
2	$P_k < P_{bad} - 1/8 \leq P_{bad} - 1/8$
	$P_k = P_{bad} - 1/8 \leq P_{bad}$



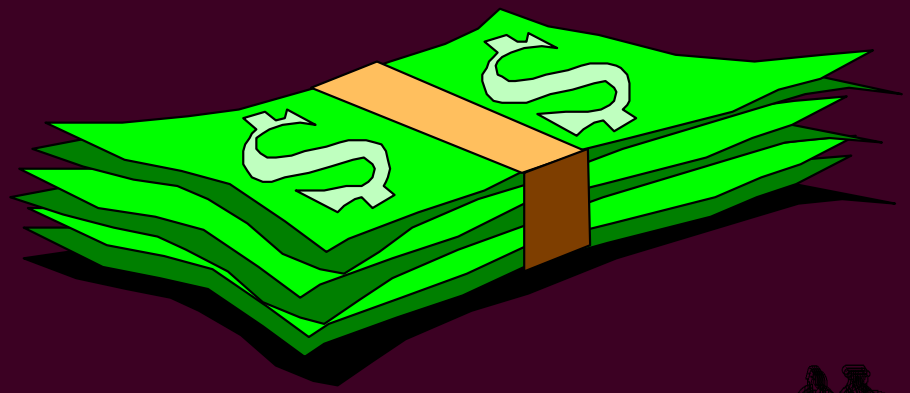

buggy entry
foothold
-1/8
q=-1 or -2

“Send More Money” Puzzle

SEND

+MORE

MONEY



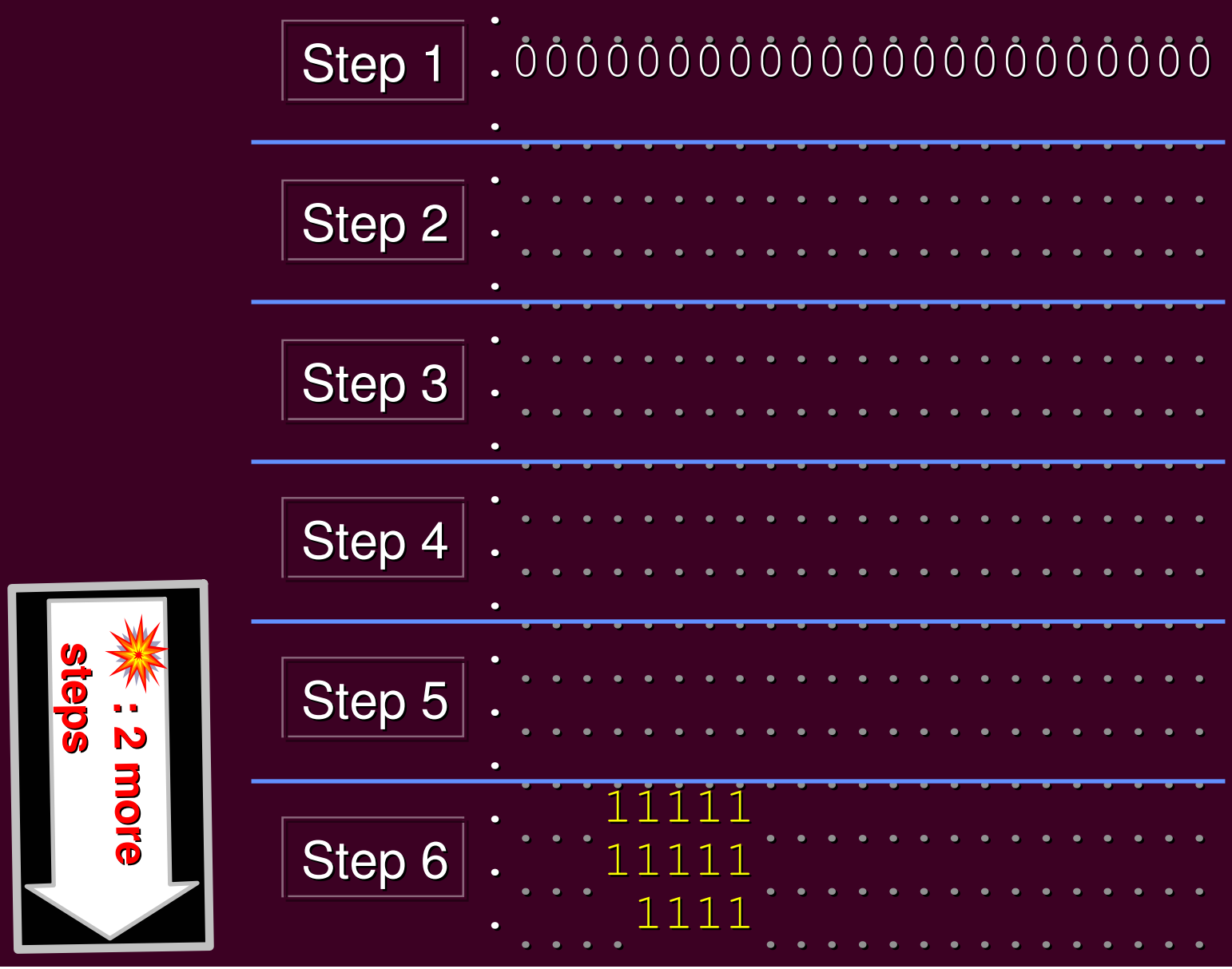
MIT

The Path to Failure

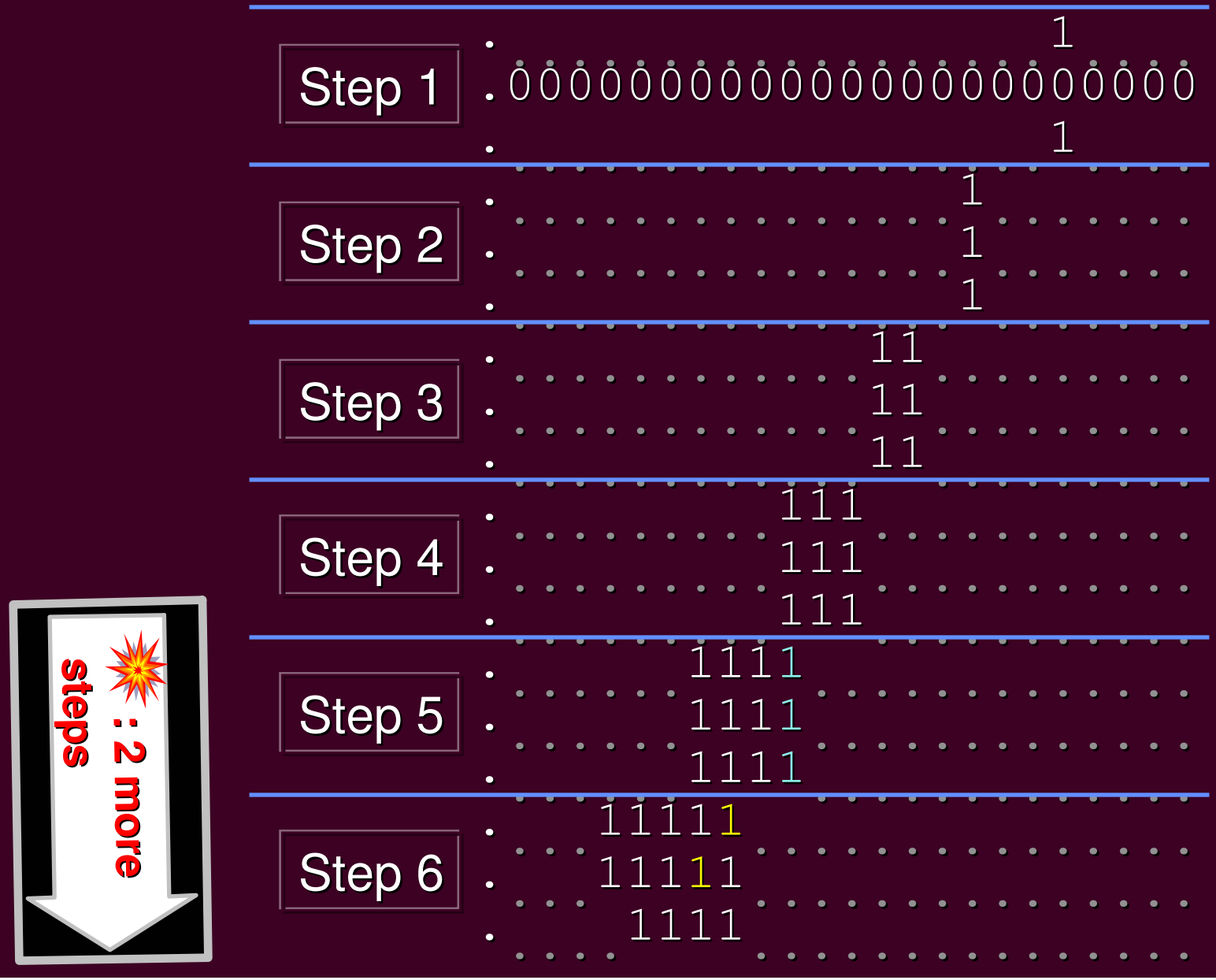
Bad Divisors: $d = 1.d_1d_2d_3d_4\underbrace{111111}_{\text{six ones}}d_{11}\ldots$

	. . . . 1 1 1 1 1
q = –2	. . . . 1 1 1 1 1
	. d_2 d_3 d_4 1 1 1 1 1
	. . 1 1 1 1 1
q = 2	. 1 1 1 1 1 1
	. $\bar{d}_2$ $\bar{d}_3$ $\bar{d}_4$ 0 0 0
	. . 0 0 0
bug	. 1 1 1 .

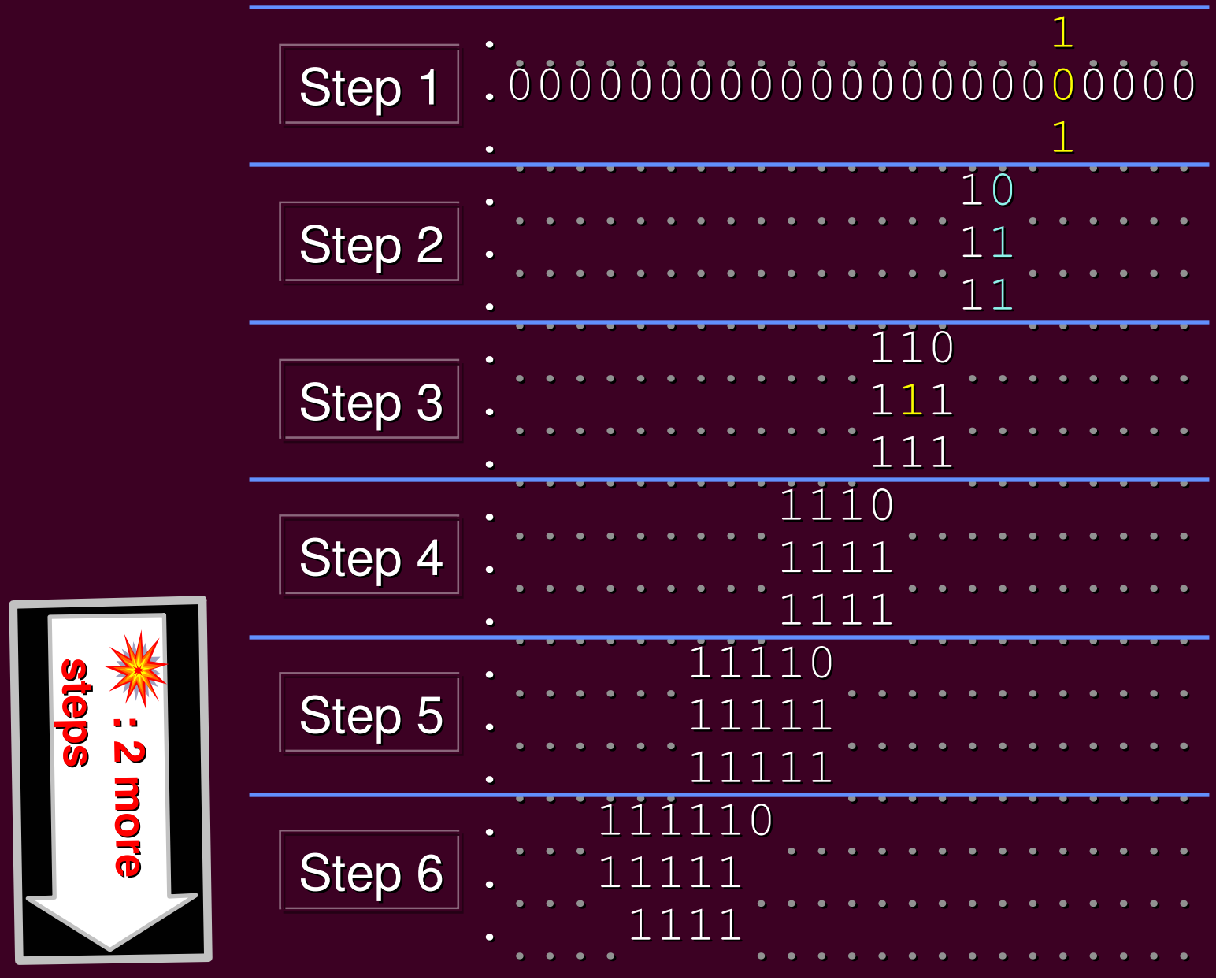
At Least Nine Steps to Failure



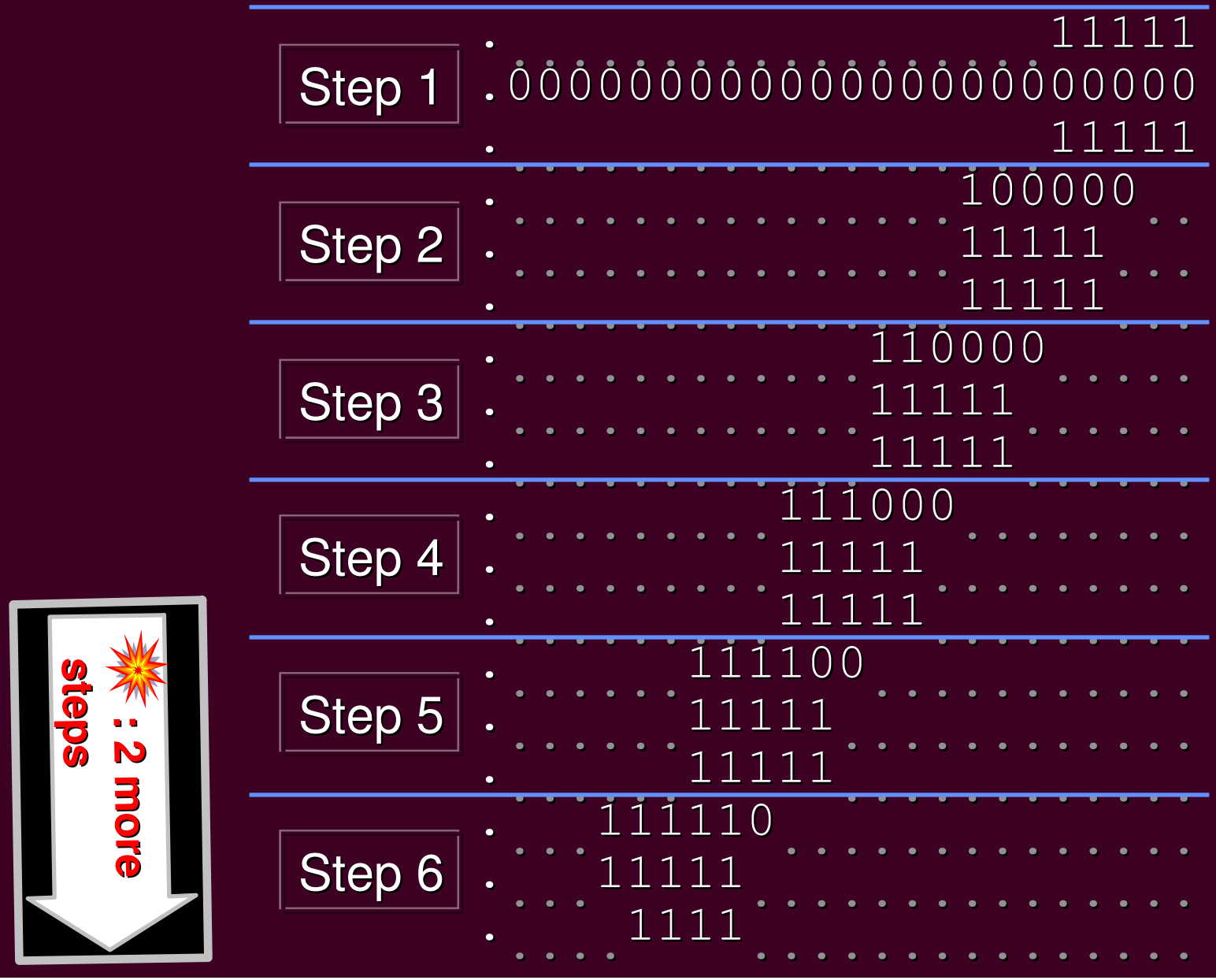
At Least Nine Steps to Failure



At Least Nine Steps to Failure



At Least Nine Steps to Failure



Conclusions

- Mathematical analysis is possible.
- Bug is more subtle and more interesting than most people realize.
- One should not be so quick to laugh at Intel's Expense.

Thanks to Teddy Slottow for his technical assistance in preparing this presentation

MIT