64-bit Integer Division for the JavaScript Platform

Arith 2026, Fulda, Germany

Sébastien Doeraene and Tobias Schlatter

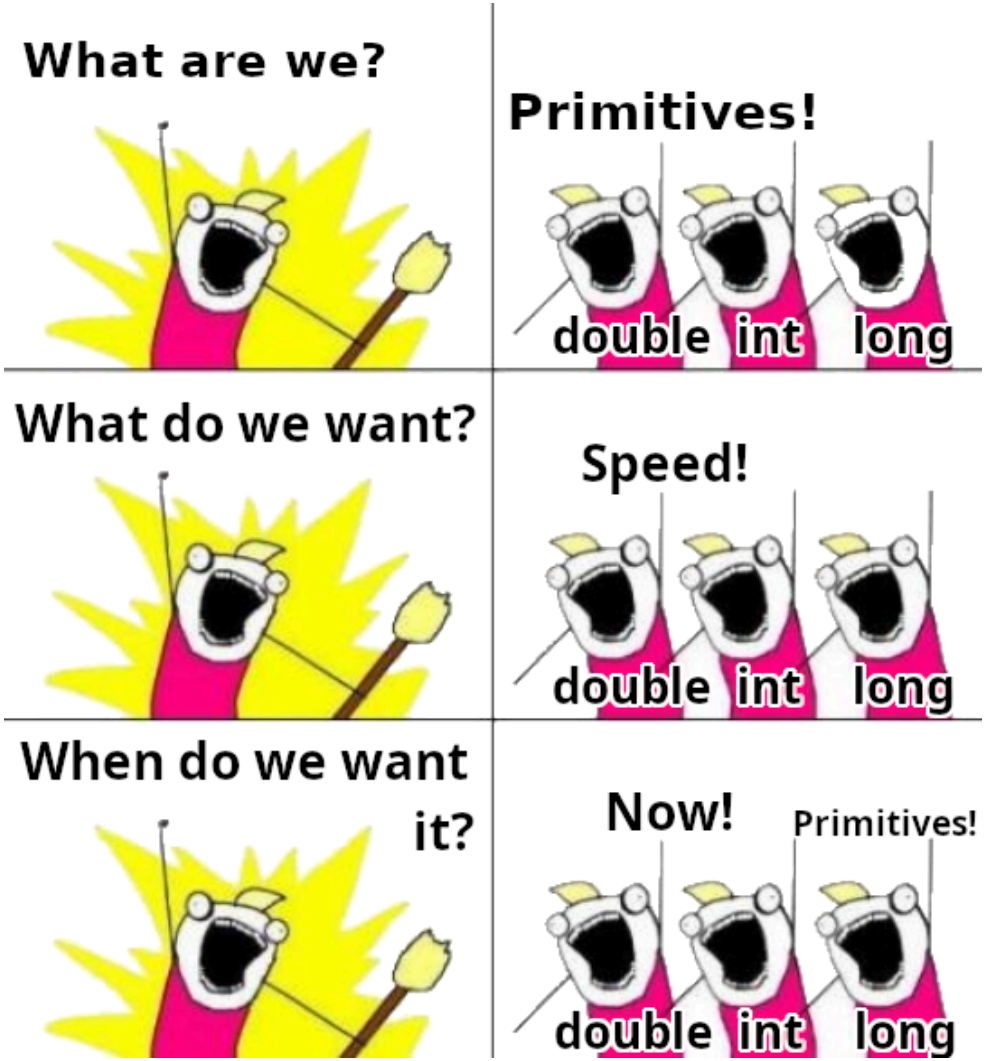
EPFL and Datahouse, Switzerland

sebastien.doeraene@epfl.ch

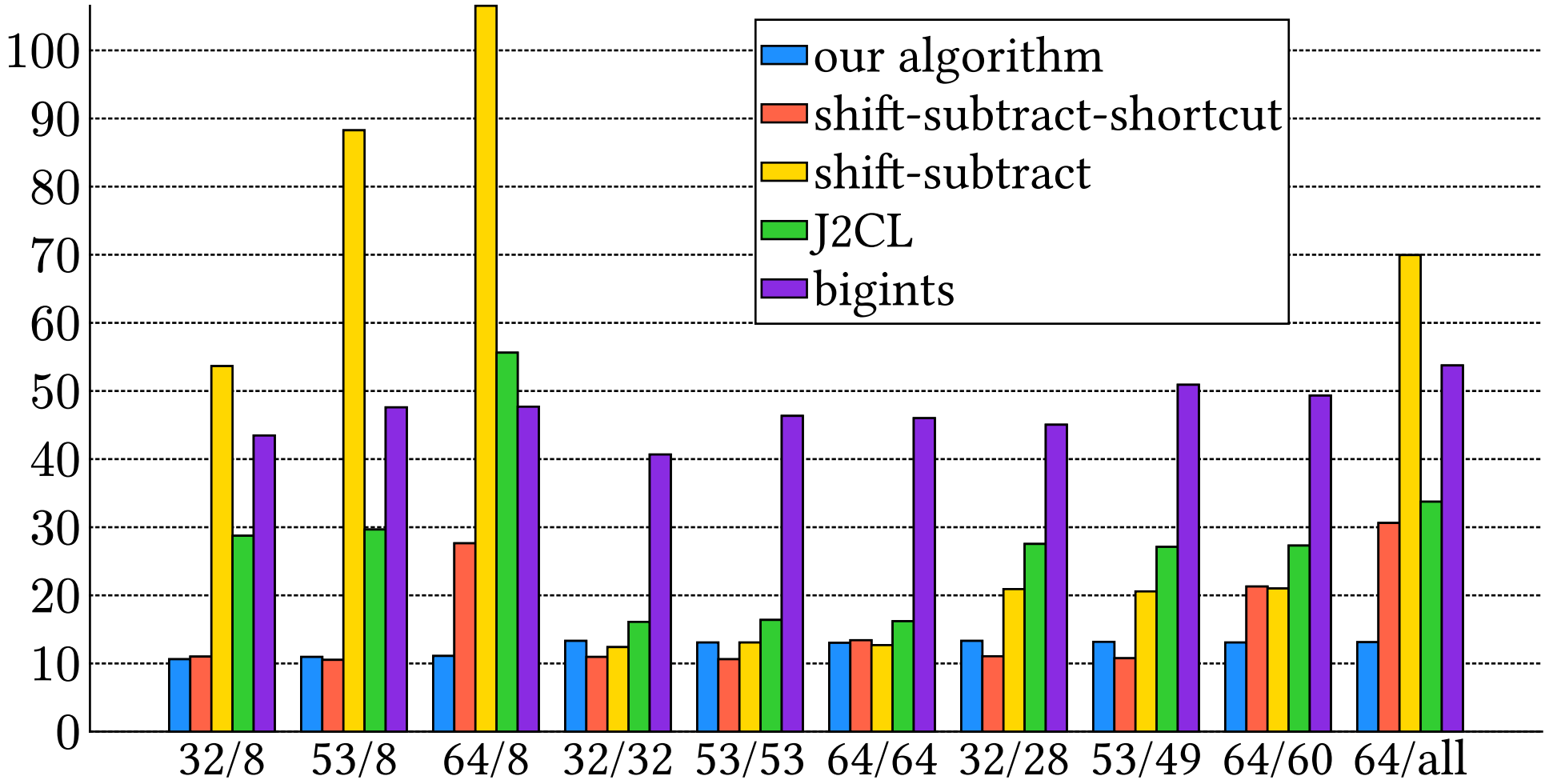
June 29th, 2026

1. Motivation

1.1 Once upon a time, in a land called “JavaScript” …



1.2 Benchmarks – normalized run time, lower is better



1.3 Alternatives

WebAssembly (Wasm)

JavaScript `bigints` for `longs`

1.3 Alternatives

WebAssembly (Wasm)

✅ Fast primitives, including `longs`

❌ Not always available

❌ Not-so-fast interop with JavaScript and the DOM

JavaScript `bigints` for `longs`

1.3 Alternatives

WebAssembly (Wasm)

✅ Fast primitives, including `longs`

❌ Not always available

❌ Not-so-fast interop with JavaScript and the DOM

JavaScript `bigints` for `longs`

✅ Easy compilation scheme

✅ Baseline since 2020

❌ Not that fast

2. Preliminaries

2.1 Notations

$\mathbb{F}_{64}$	set of binary64 values	
$\mathrm{RN} : \mathbb{R} \to \mathbb{F}_{64}$	round to nearest	
$\mathrm{div}(a, b)$	$\lfloor a/b \rfloor$	unsigned division
$\mathrm{rem}(a, b)$	$a - b \cdot \mathrm{div}(a, b)$	unsigned remainder
$a +_n b$	$a + b \bmod 2^n$	modular n-bit addition
$a -_n b$	$a - b \bmod 2^n$	
$a \times_n b$	$a \cdot b \bmod 2^n$	
$\mathrm{S}_n(a)$	$\begin{cases} a & \text{if } a < 2^{n-1} \\ a - 2^n & \text{otherwise} \end{cases}$	signed interpretation

2.2 32-bit integers in JavaScript – asm.js encoding

Abstract operation	JavaScript encoding
$a +_{32} b$	`(a+b)|0`
$a -_{32} b$	`(a-b)|0`
$a \times_{32} b$	`Math.imul(a, b)`
$\mathrm{div}(a, b)$	`((a>>>0) / (b>>>0))|0`
$\mathrm{rem}(a, b)$	`((a>>>0) % (b>>>0))|0`
$\mathrm{FP}(a)$	`a>>>0`
$\mathrm{FP}(\mathrm{S}_{32}(a))$	`a`
$\mathrm{truncate}(x) \bmod 2^{32}$	`x|0`

2.3 Basic 64-bit arithmetic

Pair of two 32-bit integers $x = (x_h, x_\ell)$

Unsigned : $x = 2^{32} x_h + x_\ell$ (by definition)

Signed : $\mathrm{S}_{64}(x) = \mathrm{S}_{64}(2^{32} x_h + x_\ell) = 2^{32} \cdot \mathrm{S}_{32}(x_h) + x_\ell$

2.3 Basic 64-bit arithmetic

Pair of two 32-bit integers $x = (x_h, x_\ell)$

Unsigned : $x = 2^{32}x_h + x_\ell$ (by definition)

Signed : $\mathrm{S}_{64}(x) = \mathrm{S}_{64}(2^{32}x_h + x_\ell) = 2^{32} \cdot \mathrm{S}_{32}(x_h) + x_\ell$

Add64

Input: $a = (a_h, a_\ell), b = (b_h, b_\ell)$
Output: $s = (s_h, s_\ell) = a +_{64} b$

$s_\ell \leftarrow a_\ell +_{32} b_\ell$
$s_h \leftarrow a_h +_{32} b_h +_{32} (s_\ell < a_\ell)$ // in JS : `((sl>>>0) < (al>>>0))|0`

2.4 Conversion to $\mathbb{F}_{64}$

2.4.1 Unsigned

$$\begin{aligned}\mathrm{RN}(x) &= \mathrm{RN}\left(2^{32}x_h + x_\ell\right) \\ &= \mathrm{RN}\left(\mathrm{RN}\left(2^{32}\cdot \mathrm{FP}(x_h)\right) + \mathrm{FP}(x_\ell)\right)\end{aligned}$$

JavaScript code: 2^{32}`*(xh>>>0) + (xl>>>0)`

2.4 Conversion to $\mathbb{F}_{64}$

2.4.1 Unsigned

$$\begin{aligned}\mathrm{RN}(x) &= \mathrm{RN}\left(2^{32}x_h + x_\ell\right)\\ &= \mathrm{RN}\left(\mathrm{RN}\left(2^{32}\cdot \mathrm{FP}(x_h)\right) + \mathrm{FP}(x_\ell)\right)\end{aligned}$$

JavaScript code: 2^{32}`*(xh>>>0) + (xl>>>0)`

2.4.2 Signed

$$\begin{aligned}\mathrm{RN}(\mathrm{S}_{64}(x)) &= \mathrm{RN}\left(2^{32}\cdot \mathrm{S}_{32}(x_h) + x_\ell\right)\\ &= \mathrm{RN}\left(\mathrm{RN}\left(2^{32}\cdot \mathrm{FP}(\mathrm{S}_{32}(x_h))\right) + \mathrm{FP}(x_\ell)\right)\end{aligned}$$

JavaScript code: 2^{32}`*xh + (xl>>>0)`

2.5 Unsigned conversion from $\mathbb{F}_{64}$

Given $x \in \mathbb{F}_{64}$, $0 \leq x < 2^{64}$, find $y = (y_h, y_\ell)$ such that $y = \lfloor x \rfloor$.

2.5 Unsigned conversion from $\mathbb{F}_{64}$

Given $x \in \mathbb{F}_{64}$, $0 \leq x < 2^{64}$, find $y = (y_h, y_\ell)$ such that $y = \lfloor x \rfloor$.

JavaScript code:

```
yh = (x*2^-32)|0
yl = x|0
```

computes:

$$y_h = \text{truncate}\big(\text{RN}\big(x \cdot 2^{-32}\big)\big) \bmod 2^{32} = \lfloor x/2^{32} \rfloor \bmod 2^{32}$$

$$y_\ell = \text{truncate}(x) \bmod 2^{32} = \lfloor x \rfloor \bmod 2^{32}$$

3. The divison algorithm

3.1 Unsigned division

Given $a = (a_h, a_\ell)$, $b = (b_h, b_\ell)$, $b \neq 0$,
find $q = (q_h, q_\ell) = \mathrm{div}(a, b)$ and $r = (r_h, r_\ell) = \mathrm{rem}(a, b)$

3.2 Three cases

Div64

Input: $a = (a_h, a_\ell)$, $b = (b_h, b_\ell)$, $b \neq 0$
Output: $q = (q_h, q_\ell) = \mathrm{div}(a, b)$, $r = (r_h, r_\ell) = \mathrm{rem}(a, b)$

if $b < 2^{21}$ **then**

else if $b < 2^{63}$ **then**

else

end

3.2 Three cases

Div64

Input: $a = (a_h, a_\ell)$, $b = (b_h, b_\ell)$, $b \neq 0$
Output: $q = (q_h, q_\ell) = \mathrm{div}(a, b)$, $r = (r_h, r_\ell) = \mathrm{rem}(a, b)$

if $b < 2^{21}$ **then**

else if $b < 2^{63}$ **then**

else
  if $a \geq b$ **then**
    $q \leftarrow 1 \;\; ; \;\; r \leftarrow a -_{64} b$
  else
    $q \leftarrow 0 \;\; ; \;\; r \leftarrow a$
  end
end

3.2 Three cases

Div64

Input: $a = (a_h, a_\ell)$, $b = (b_h, b_\ell)$, $b \neq 0$
Output: $q = (q_h, q_\ell) = \mathrm{div}(a, b)$, $r = (r_h, r_\ell) = \mathrm{rem}(a, b)$

```
if b < 2^21 then
    (q, r) ← Div64-Case1(a, b)
else if b < 2^63 then
    (q, r) ← Div64-Case2(a, b)
else
    if a ≥ b then
        q ← 1  ;  r ← a −_64 b
    else
        q ← 0  ;  r ← a
    end
end
```

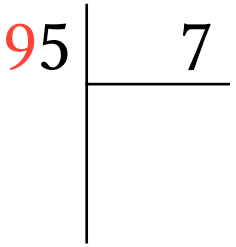
3.3 Unsigned division, $0 < b < 2^{21}$, for humans

As a human, how do you divide 95 by 7?

$$\begin{array}{r|l} 95 & 7 \\ \hline & \\ & \end{array}$$

3.3 Unsigned division, $0 < b < 2^{21}$, for humans

As a human, how do you divide 95 by 7?



3.3 Unsigned division, $0 < b < 2^{21}$, for humans

As a human, how do you divide 95 by 7?

$$\begin{array}{r|l} 95 & 7 \\ \hline 1 & \end{array}$$

3.3 Unsigned division, $0 < b < 2^{21}$, for humans

As a human, how do you divide 95 by 7?

$$\begin{array}{r|l} 95 & 7 \\ \hline 2 & 1 \end{array}$$

3.3 Unsigned division, $0 < b < 2^{21}$, for humans

As a human, how do you divide 95 by 7?

$$\begin{array}{r|l} 95 & 7 \\ \hline 25 & 1 \\ \\ \end{array}$$

3.3 Unsigned division, $0 < b < 2^{21}$, for humans

As a human, how do you divide 95 by 7?

95	7
25	1

3.3 Unsigned division, $0 < b < 2^{21}$, for humans

As a human, how do you divide 95 by 7?

95	7
25	13

3.3 Unsigned division, $0 < b < 2^{21}$, for humans

As a human, how do you divide 95 by 7?

$$\begin{array}{r|r} 95 & 7 \\ \hline 25 & 13 \\ 4 & \end{array}$$

3.3 Unsigned division, $0 < b < 2^{21}$, for humans

As a human, how do you divide 95 by 7?

95	7
25	13
4	

Quotient: 13

Remainder: 4

3.4 Unsigned division, $0 < b < 2^{21}$, for the machine

Same thing, but in base 2^{32}

Div64-Case1

Input: $a = (a_h, a_\ell)$, $b = (b_h, b_\ell)$, $0 < b < 2^{21}$

Output: $q = (q_h, q_\ell) = \mathrm{div}(a, b)$, $r = (r_h, r_\ell) = \mathrm{rem}(a, b)$

Assert: $b_h = 0$

$q_h \leftarrow \mathrm{div}(a_h, b_\ell)$

$k \leftarrow a_h -_{32} (b_\ell \times_{32} q_h)$

$a' \leftarrow (k, a_\ell) = 2^{32}k + a_\ell$

$q_0 \leftarrow \mathrm{RN}(\mathrm{RN}(a') / \mathrm{FP}(b_\ell))$

Assert: $0 \le q_0 < 2^{32}$

$q_\ell \leftarrow \lfloor q_0 \rfloor$

$r_h \leftarrow 0$

$r_\ell \leftarrow a_\ell -_{32} (b_\ell \times_{32} q_\ell)$

$a_h\ a_\ell \mid b_\ell$

3.4 Unsigned division, $0 < b < 2^{21}$, for the machine

Same thing, but in base 2^{32}

Div64-Case1

Input: $a = (a_h, a_\ell)$, $b = (b_h, b_\ell)$, $0 < b < 2^{21}$

Output: $q = (q_h, q_\ell) = \mathrm{div}(a, b)$, $r = (r_h, r_\ell) = \mathrm{rem}(a, b)$

Assert: $b_h = 0$

$q_h \leftarrow \mathrm{div}(a_h, b_\ell)$

$k \leftarrow a_h -_{32} (b_\ell \times_{32} q_h)$

$a' \leftarrow (k, a_\ell) = 2^{32}k + a_\ell$

$q_0 \leftarrow \mathrm{RN}(\mathrm{RN}(a') / \mathrm{FP}(b_\ell))$

Assert: $0 \le q_0 < 2^{32}$

$q_\ell \leftarrow \lfloor q_0 \rfloor$

$r_h \leftarrow 0$

$r_\ell \leftarrow a_\ell -_{32} (b_\ell \times_{32} q_\ell)$

$a_h \; a_\ell \;|\; b_\ell$

3.4 Unsigned division, $0 < b < 2^{21}$, for the machine

Same thing, but in base 2^{32}

Div64-Case1

Input: $a = (a_h, a_\ell)$, $b = (b_h, b_\ell)$, $0 < b < 2^{21}$

Output: $q = (q_h, q_\ell) = \mathrm{div}(a, b)$, $r = (r_h, r_\ell) = \mathrm{rem}(a, b)$

Assert: $b_h = 0$

$q_h \leftarrow \mathrm{div}(a_h, b_\ell)$

$k \leftarrow a_h -_{32} (b_\ell \times_{32} q_h)$

$a' \leftarrow (k, a_\ell) = 2^{32}k + a_\ell$

$q_0 \leftarrow \mathrm{RN}(\mathrm{RN}(a') / \mathrm{FP}(b_\ell))$

Assert: $0 \le q_0 < 2^{32}$

$q_\ell \leftarrow \lfloor q_0 \rfloor$

$r_h \leftarrow 0$

$r_\ell \leftarrow a_\ell -_{32} (b_\ell \times_{32} q_\ell)$

$a_h \; a_\ell \;|\; b_\ell$

q_h

3.4 Unsigned division, $0 < b < 2^{21}$, for the machine

Same thing, but in base 2^{32}

Div64-Case1

Input: $a = (a_h, a_\ell)$, $b = (b_h, b_\ell)$, $0 < b < 2^{21}$

Output: $q = (q_h, q_\ell) = \mathrm{div}(a, b)$, $r = (r_h, r_\ell) = \mathrm{rem}(a, b)$

Assert: $b_h = 0$

$q_h \leftarrow \mathrm{div}(a_h, b_\ell)$

$k \leftarrow a_h -_{32} (b_\ell \times_{32} q_h)$

$a' \leftarrow (k, a_\ell) = 2^{32}k + a_\ell$

$q_0 \leftarrow \mathrm{RN}(\mathrm{RN}(a') / \mathrm{FP}(b_\ell))$

Assert: $0 \le q_0 < 2^{32}$

$q_\ell \leftarrow \lfloor q_0 \rfloor$

$r_h \leftarrow 0$

$r_\ell \leftarrow a_\ell -_{32} (b_\ell \times_{32} q_\ell)$

$a_h \; a_\ell \mid b_\ell$

$k \mid q_h$

3.4 Unsigned division, $0 < b < 2^{21}$, for the machine

Same thing, but in base 2^{32}

Div64-Case1

Input: $a = (a_h, a_\ell)$, $b = (b_h, b_\ell)$, $0 < b < 2^{21}$

Output: $q = (q_h, q_\ell) = \mathrm{div}(a, b)$, $r = (r_h, r_\ell) = \mathrm{rem}(a, b)$

Assert: $b_h = 0$

$q_h \leftarrow \mathrm{div}(a_h, b_\ell)$

$k \leftarrow a_h -_{32} (b_\ell \times_{32} q_h)$

$a' \leftarrow (k, a_\ell) = 2^{32}k + a_\ell$

$q_0 \leftarrow \mathrm{RN}(\mathrm{RN}(a') / \mathrm{FP}(b_\ell))$

Assert: $0 \le q_0 < 2^{32}$

$q_\ell \leftarrow \lfloor q_0 \rfloor$

$r_h \leftarrow 0$

$r_\ell \leftarrow a_\ell -_{32} (b_\ell \times_{32} q_\ell)$

$a_h\ a_\ell$	b_ℓ
$k\ a_\ell$	q_h

3.4 Unsigned division, $0 < b < 2^{21}$, for the machine

Same thing, but in base 2^{32}

Div64-Case1

Input: $a = (a_h, a_\ell)$, $b = (b_h, b_\ell)$, $0 < b < 2^{21}$

Output: $q = (q_h, q_\ell) = \mathrm{div}(a, b)$, $r = (r_h, r_\ell) = \mathrm{rem}(a, b)$

Assert: $b_h = 0$

$q_h \leftarrow \mathrm{div}(a_h, b_\ell)$

$k \leftarrow a_h -_{32} (b_\ell \times_{32} q_h)$

$a' \leftarrow (k, a_\ell) = 2^{32}k + a_\ell$

$q_0 \leftarrow \mathrm{RN}(\mathrm{RN}(a')/\mathrm{FP}(b_\ell))$

Assert: $0 \le q_0 < 2^{32}$

$q_\ell \leftarrow \lfloor q_0 \rfloor$

$r_h \leftarrow 0$

$r_\ell \leftarrow a_\ell -_{32} (b_\ell \times_{32} q_\ell)$

$$\begin{array}{rl|l} & a_h\ a_\ell & b_\ell \\ \hline a' = & k\ a_\ell & q_h \end{array}$$

3.4 Unsigned division, $0 < b < 2^{21}$, for the machine

Same thing, but in base 2^{32}

Div64-Case1

Input: $a = (a_h, a_\ell)$, $b = (b_h, b_\ell)$, $0 < b < 2^{21}$

Output: $q = (q_h, q_\ell) = \mathrm{div}(a, b)$, $r = (r_h, r_\ell) = \mathrm{rem}(a, b)$

Assert: $b_h = 0$

$q_h \leftarrow \mathrm{div}(a_h, b_\ell)$

$k \leftarrow a_h -_{32} (b_\ell \times_{32} q_h)$

$a' \leftarrow (k, a_\ell) = 2^{32}k + a_\ell$

$q_0 \leftarrow \mathrm{RN}(\mathrm{RN}(a')/\mathrm{FP}(b_\ell))$

Assert: $0 \le q_0 < 2^{32}$

$q_\ell \leftarrow \lfloor q_0 \rfloor$

$r_h \leftarrow 0$

$r_\ell \leftarrow a_\ell -_{32} (b_\ell \times_{32} q_\ell)$

$$\begin{array}{rl|r} & a_h\ a_\ell & b_\ell \\ \hline a' = & k\ a_\ell & q_h\ q_\ell \end{array}$$

3.4 Unsigned division, $0 < b < 2^{21}$, for the machine

Same thing, but in base 2^{32}

Div64-Case1

Input: $a = (a_h, a_\ell)$, $b = (b_h, b_\ell)$, $0 < b < 2^{21}$

Output: $q = (q_h, q_\ell) = \mathrm{div}(a, b)$, $r = (r_h, r_\ell) = \mathrm{rem}(a, b)$

Assert: $b_h = 0$

$q_h \leftarrow \mathrm{div}(a_h, b_\ell)$

$k \leftarrow a_h -_{32} (b_\ell \times_{32} q_h)$

$a' \leftarrow (k, a_\ell) = 2^{32}k + a_\ell$

$q_0 \leftarrow \mathrm{RN}(\mathrm{RN}(a')/\mathrm{FP}(b_\ell))$

Assert: $0 \le q_0 < 2^{32}$

$q_\ell \leftarrow \lfloor q_0 \rfloor$

$r_h \leftarrow 0$

$r_\ell \leftarrow a_\ell -_{32} (b_\ell \times_{32} q_\ell)$

$$\begin{array}{rr|r} & a_h\ a_\ell & b_\ell \\ \hline a' = & k\ a_\ell & q_h\ q_\ell \\ & r_\ell & \end{array}$$

3.4 Unsigned division, $0 < b < 2^{21}$, for the machine

Same thing, but in base 2^{32}

Div64-Case1

Input: $a = (a_h, a_\ell)$, $b = (b_h, b_\ell)$, $0 < b < 2^{21}$

Output: $q = (q_h, q_\ell) = \mathrm{div}(a, b)$, $r = (r_h, r_\ell) = \mathrm{rem}(a, b)$

Assert: $b_h = 0$

$q_h \leftarrow \mathrm{div}(a_h, b_\ell)$

$k \leftarrow a_h -_{32} (b_\ell \times_{32} q_h)$

$a' \leftarrow (k, a_\ell) = 2^{32}k + a_\ell$

$q_0 \leftarrow \mathrm{RN}(\mathrm{RN}(a')/\mathrm{FP}(b_\ell))$

Assert: $0 \le q_0 < 2^{32}$

$q_\ell \leftarrow \lfloor q_0 \rfloor$

$r_h \leftarrow 0$

$r_\ell \leftarrow a_\ell -_{32} (b_\ell \times_{32} q_\ell)$

$$\begin{array}{rr|l} & a_h\ a_\ell & b_\ell \\ \hline a' = & k\ a_\ell & q_h\ q_\ell \\ & r_\ell & \end{array}$$

Quotient: $q_h\ q_\ell$

Remainder: $0\ r_\ell$

3.5 Dividing a' by b_ℓ

$$q_0 \leftarrow \mathrm{RN}(\mathrm{RN}(a') / \mathrm{FP}(b_\ell))$$
Assert: $0 \leq q_0 < 2^{32}$
$$q_\ell \leftarrow \lfloor q_0 \rfloor$$

Is it true that $q_\ell = \mathrm{div}(a', b_\ell)$?

3.5 Dividing a' by b_ℓ

$$q_0 \leftarrow \mathrm{RN}(\mathrm{RN}(a') / \mathrm{FP}(b_\ell))$$
Assert: $0 \leq q_0 < 2^{32}$
$$q_\ell \leftarrow \lfloor q_0 \rfloor$$

Is it true that $q_\ell = \mathrm{div}(a', b_\ell)$?

$a' = 2^{32}k + a_\ell$, and $k < b_\ell$, so $a' < 2^{32} b_\ell < 2^{53}$

3.5 Dividing a' by b_ℓ

$q_0 \leftarrow \mathrm{RN}(\mathrm{RN}(a')/\,\mathrm{FP}(b_\ell))$
Assert: $0 \le q_0 < 2^{32}$
$q_\ell \leftarrow \lfloor q_0 \rfloor$

Is it true that $q_\ell = \mathrm{div}(a', b_\ell)$?

$a' = 2^{32}k + a_\ell$, and $k < b_\ell$, so $a' < 2^{32}b_\ell < 2^{53}$

$q_0 = \mathrm{RN}(\mathrm{RN}(a')/\,\mathrm{FP}(b_\ell)) = \mathrm{RN}(a'/b_\ell)$

$q_\ell = \lfloor \mathrm{RN}(a'/b_\ell) \rfloor$

3.5 Dividing a' by b_ℓ

$q_0 \leftarrow \mathrm{RN}(\mathrm{RN}(a')/\,\mathrm{FP}(b_\ell))$
Assert: $0 \le q_0 < 2^{32}$
$q_\ell \leftarrow \lfloor q_0 \rfloor$

Is it true that $q_\ell = \mathrm{div}(a', b_\ell)$?

$a' = 2^{32}k + a_\ell$, and $k < b_\ell$, so $a' < 2^{32}b_\ell < 2^{53}$

$q_0 = \mathrm{RN}(\mathrm{RN}(a')/\,\mathrm{FP}(b_\ell)) = \mathrm{RN}(a'/b_\ell)$

$q_\ell = \lfloor \mathrm{RN}(a'/b_\ell) \rfloor = \mathrm{div}(a', b_\ell)$ ✅

because of Theorem 1 in “The Euclidean Division Implemented with a Floating-Point Division and a Floor” by Vincent Lefèvre.

3.6 Unsigned division, $2^{21} \leq b \leq 2^{63}$, for humans

As a human, how do you divide $a = 857$ by $b = 31$?

3.6 Unsigned division, $2^{21} \leq b \leq 2^{63}$, for humans

As a human, how do you divide $a = 857$ by $b = 31$?

$a \approx 850 = \hat{a}$ $\qquad$ $b \approx 30 = \hat{b}$

3.6 Unsigned division, $2^{21} \leq b \leq 2^{63}$, for humans

As a human, how do you divide $a = 857$ by $b = 31$?

$a \approx 850 = \hat{a}$ $\qquad$ $b \approx 30 = \hat{b}$

$\hat{a}/\hat{b} = 850/30 = 85/3$

3.6 Unsigned division, $2^{21} \leq b \leq 2^{63}$, for humans

As a human, how do you divide $a = 857$ by $b = 31$?

$a \approx 850 = \hat{a}$ $\qquad$ $b \approx 30 = \hat{b}$

$\hat{a}/\hat{b} = 850/30 = 85/3 =$ 🤔

3.6 Unsigned division, $2^{21} \leq b \leq 2^{63}$, for humans

As a human, how do you divide $a = 857$ by $b = 31$?

$$a \approx 850 = \hat{a} \qquad b \approx 30 = \hat{b}$$

$$\hat{a}/\hat{b} = 850/30 = 85/3 = (60 + 24 + 1)/3 \approx 28.3 = \hat{q}_0$$

3.6 Unsigned division, $2^{21} \leq b \leq 2^{63}$, for humans

As a human, how do you divide $a = 857$ by $b = 31$?

$a \approx 850 = \hat{a} \qquad b \approx 30 = \hat{b}$

$\hat{a}/\hat{b} = 850/30 = 85/3 = (60 + 24 + 1)/3 \approx 28.3 = \hat{q}_0$

$\hat{q} = \lfloor \hat{q}_0 \rfloor = \lfloor 28.3 \rfloor = 28$

3.6 Unsigned division, $2^{21} \leq b \leq 2^{63}$, for humans

As a human, how do you divide $a = 857$ by $b = 31$?

$$a \approx 850 = \hat{a} \qquad b \approx 30 = \hat{b}$$

$$\hat{a}/\hat{b} = 850/30 = 85/3 = (60 + 24 + 1)/3 \approx 28.3 = \hat{q}_0$$

$$\hat{q} = \lfloor \hat{q}_0 \rfloor = \lfloor 28.3 \rfloor = 28$$

$$\hat{r} = a - b \cdot \hat{q} = 857 - 31 \cdot 28 = -11$$

3.6 Unsigned division, $2^{21} \leq b \leq 2^{63}$, for humans

As a human, how do you divide $a = 857$ by $b = 31$?

$$a \approx 850 = \hat{a} \qquad b \approx 30 = \hat{b}$$

$$\hat{a}/\hat{b} = 850/30 = 85/3 = (60 + 24 + 1)/3 \approx 28.3 = \hat{q}_0$$

$$\hat{q} = \lfloor \hat{q}_0 \rfloor = \lfloor 28.3 \rfloor = 28$$

$$\hat{r} = a - b \cdot \hat{q} = 857 - 31 \cdot 28 = -11$$

Oops, let's correct that:

$$q = \hat{q} - 1 = 27$$

$$r = \hat{r} + b = 20$$

3.7 Unsigned division, $2^{21} \leq b \leq 2^{63}$, for machines

Div64-Case2 (spoiler: wrong version)

Input: $a = (a_h, a_\ell)$, $b = (b_h, b_\ell)$, $2^{21} \leq b \leq 2^{63}$
Output: $q = (q_h, q_\ell) = \mathrm{div}(a, b)$, $r = (r_h, r_\ell) = \mathrm{rem}(a, b)$

$\hat{q}_0 \leftarrow \mathrm{RN}(\mathrm{RN}(a) / \mathrm{RN}(b))$
Assert: $0 \leq \hat{q}_0 < 2^{64}$
$\hat{q} = (\hat{q}_h, \hat{q}_\ell) \leftarrow \lfloor \hat{q}_0 \rfloor$
$\hat{r} = (\hat{r}_h, \hat{r}_\ell) \leftarrow a -_{64} (b \times_{64} \hat{q})$
if $\mathrm{S}_{64}(\hat{r}) < 0$ **then**
  $q \leftarrow \hat{q} -_{64} 1$
  $r \leftarrow \hat{r} +_{64} b$
else
  $q \leftarrow \hat{q}$
  $r \leftarrow \hat{r}$
end

3.7 Unsigned division, $2^{21} \leq b \leq 2^{63}$, for machines

Div64-Case2 (spoiler: wrong version)

Input: $a = (a_h, a_\ell)$, $b = (b_h, b_\ell)$, $2^{21} \leq b \leq 2^{63}$
Output: $q = (q_h, q_\ell) = \mathrm{div}(a, b)$, $r = (r_h, r_\ell) = \mathrm{rem}(a, b)$

$\hat{q}_0 \leftarrow \mathrm{RN}(\mathrm{RN}(a) / \mathrm{RN}(b))$
Assert: $0 \leq \hat{q}_0 < 2^{64}$
$\hat{q} = (\hat{q}_h, \hat{q}_\ell) \leftarrow \lfloor \hat{q}_0 \rfloor$
$\hat{r} = (\hat{r}_h, \hat{r}_\ell) \leftarrow a -_{64} (b \times_{64} \hat{q})$
if $\mathrm{S}_{64}(\hat{r}) < 0$ **then**
  $q \leftarrow \hat{q} -_{64} 1$
  $r \leftarrow \hat{r} +_{64} b$
else
  $q \leftarrow \hat{q}$
  $r \leftarrow \hat{r}$
end

3.7 Unsigned division, $2^{21} \leq b \leq 2^{63}$, for machines

Div64-Case2 (spoiler: wrong version)

Input: $a = (a_h, a_\ell)$, $b = (b_h, b_\ell)$, $2^{21} \leq b \leq 2^{63}$
Output: $q = (q_h, q_\ell) = \mathrm{div}(a, b)$, $r = (r_h, r_\ell) = \mathrm{rem}(a, b)$

$\hat{q}_0 \leftarrow \mathrm{RN}(\mathrm{RN}(a)/\,\mathrm{RN}(b))$
Assert: $0 \leq \hat{q}_0 < 2^{64}$
$\hat{q} = (\hat{q}_h, \hat{q}_\ell) \leftarrow \lfloor \hat{q}_0 \rfloor$
$\hat{r} = (\hat{r}_h, \hat{r}_\ell) \leftarrow a -_{64} (b \times_{64} \hat{q})$
if $\mathrm{S}_{64}(\hat{r}) < 0$ **then**
  $q \leftarrow \hat{q} -_{64} 1$
  $r \leftarrow \hat{r} +_{64} b$
else
  $q \leftarrow \hat{q}$
  $r \leftarrow \hat{r}$
end

3.7 Unsigned division, $2^{21} \leq b \leq 2^{63}$, for machines

Div64-Case2 (spoiler: wrong version)

Input: $a = (a_h, a_\ell)$, $b = (b_h, b_\ell)$, $2^{21} \leq b \leq 2^{63}$
Output: $q = (q_h, q_\ell) = \mathrm{div}(a, b)$, $r = (r_h, r_\ell) = \mathrm{rem}(a, b)$

$\hat{q}_0 \leftarrow \mathrm{RN}(\mathrm{RN}(a)/\,\mathrm{RN}(b))$
Assert: $0 \leq \hat{q}_0 < 2^{64}$
$\hat{q} = (\hat{q}_h, \hat{q}_\ell) \leftarrow \lfloor \hat{q}_0 \rfloor$
$\hat{r} = (\hat{r}_h, \hat{r}_\ell) \leftarrow a -_{64} (b \times_{64} \hat{q})$
if $\mathrm{S}_{64}(\hat{r}) < 0$ **then**
  $q \leftarrow \hat{q} -_{64} 1$
  $r \leftarrow \hat{r} +_{64} b$
else
  $q \leftarrow \hat{q}$
  $r \leftarrow \hat{r}$
end

3.8 What can we say about $\hat{q}$?

[1]

$$\hat{a} \leftarrow \mathrm{RN}(a)$$
$$\hat{b} \leftarrow \mathrm{RN}(b)$$
$$\hat{q}_0 \leftarrow \mathrm{RN}\left(\hat{a}/\hat{b}\right)$$
$$\hat{q} = (\hat{q}_h, \hat{q}_\ell) \leftarrow \lfloor \hat{q}_0 \rfloor$$

Is it true that $0 \leq \hat{q} - \mathrm{div}(a, b) \leq 1$?

---

[1]

3.8 What can we say about $\hat{q}$?

$$\hat{a} \leftarrow \mathrm{RN}(a)$$
$$\hat{b} \leftarrow \mathrm{RN}(b)$$
$$\hat{q}_0 \leftarrow \mathrm{RN}\left(\hat{a}/\hat{b}\right)$$
$$\hat{q} = (\hat{q}_h, \hat{q}_\ell) \leftarrow \lfloor \hat{q}_0 \rfloor$$

Is it true that $0 \leq \hat{q} - \mathrm{div}(a, b) \leq 1$?

Property[1]:

$$\left| \frac{\mathrm{RN}\left(\hat{a}/\hat{b}\right) - a/b}{a/b} \right| < 2^{-51}$$

[1] "Error in ulps of the multiplication or division by a correctly-rounded function or constant in binary floating-point arithmetic" by N. Brisebarre, J.-M. Muller, and J. Picot.

3.8 What can we say about $\hat{q}$?

$$\hat{a} \leftarrow \mathrm{RN}(a)$$
$$\hat{b} \leftarrow \mathrm{RN}(b)$$
$$\hat{q}_0 \leftarrow \mathrm{RN}\left(\hat{a}/\hat{b}\right)$$
$$\hat{q} = (\hat{q}_h, \hat{q}_\ell) \leftarrow \lfloor \hat{q}_0 \rfloor$$

Is it true that $0 \leq \hat{q} - \mathrm{div}(a, b) \leq 1$?

Property[1]:

$$\left| \frac{\mathrm{RN}\left(\hat{a}/\hat{b}\right) - a/b}{a/b} \right| < 2^{-51}$$

and (assuming $a \neq 0$)
$2^{-63} \leq a/b,\ \hat{q}_0 \leq 2^{43}$

[1] "Error in ulps of the multiplication or division by a correctly-rounded function or constant in binary floating-point arithmetic" by N. Brisebarre, J.-M. Muller, and J. Picot.

3.8 What can we say about $\hat{q}$?

$$\hat{a} \leftarrow \mathrm{RN}(a)$$
$$\hat{b} \leftarrow \mathrm{RN}(b)$$
$$\hat{q}_0 \leftarrow \mathrm{RN}\left(\hat{a}/\hat{b}\right)$$
$$\hat{q} = (\hat{q}_h, \hat{q}_\ell) \leftarrow \lfloor \hat{q}_0 \rfloor$$

Is it true that $0 \leq \hat{q} - \mathrm{div}(a, b) \leq 1$?

$$|\hat{q}_0 - a/b| < 2^{-51} \cdot a/b < 2^{-51} \cdot 2^{43} = 2^{-8}$$

Property[1]:

$$\left| \frac{\mathrm{RN}\left(\hat{a}/\hat{b}\right) - a/b}{a/b} \right| < 2^{-51}$$

and (assuming $a \neq 0$)
$2^{-63} \leq a/b$, $\hat{q}_0 \leq 2^{43}$

[1] "Error in ulps of the multiplication or division by a correctly-rounded function or constant in binary floating-point arithmetic" by N. Brisebarre, J.-M. Muller, and J. Picot.

3.8 What can we say about $\hat{q}$?

$$\hat{a} \leftarrow \mathrm{RN}(a)$$
$$\hat{b} \leftarrow \mathrm{RN}(b)$$
$$\hat{q}_0 \leftarrow \mathrm{RN}\left(\hat{a}/\hat{b}\right)$$
$$\hat{q} = (\hat{q}_h, \hat{q}_\ell) \leftarrow \lfloor \hat{q}_0 \rfloor$$

Is it true that $0 \leq \hat{q} - \mathrm{div}(a, b) \leq 1$?

$$|\hat{q}_0 - a/b| < 2^{-51} \cdot a/b < 2^{-51} \cdot 2^{43} = 2^{-8}$$

$$-2^{-8} < \hat{q}_0 - a/b < 2^{-8}$$ ❌ no good :-(

Property[1]:

$$\left| \frac{\mathrm{RN}\left(\hat{a}/\hat{b}\right) - a/b}{a/b} \right| < 2^{-51}$$

and (assuming $a \neq 0$)
$2^{-63} \leq a/b,\ \hat{q}_0 \leq 2^{43}$

[1] "Error in ulps of the multiplication or division by a correctly-rounded function or constant in binary floating-point arithmetic" by N. Brisebarre, J.-M. Muller, and J. Picot.

3.9 Fixed computation

$$\hat{a} \leftarrow \mathrm{RN}(a)$$
$$\hat{b} \leftarrow \mathrm{RN}(b)$$
$$\hat{q}_0 \leftarrow \mathrm{RN}\left(\hat{a}/\hat{b}\right)$$
$$\hat{q}_1 \leftarrow \mathrm{RN}(\hat{q}_0 + 2^{-8})$$
$$\hat{q} = (\hat{q}_h, \hat{q}_\ell) \leftarrow \lfloor \hat{q}_1 \rfloor$$

Is it true that $0 \leq \hat{q} - \mathrm{div}(a, b) \leq 1$?

$$-2^{-8} < \hat{q}_0 - a/b < 2^{-8}$$

3.9 Fixed computation

$$\hat{a} \leftarrow \mathrm{RN}(a)$$
$$\hat{b} \leftarrow \mathrm{RN}(b)$$
$$\hat{q}_0 \leftarrow \mathrm{RN}\left(\hat{a}/\hat{b}\right)$$
$$\hat{q}_1 \leftarrow \mathrm{RN}(\hat{q}_0 + 2^{-8})$$
$$\hat{q} = (\hat{q}_h, \hat{q}_\ell) \leftarrow \lfloor \hat{q}_1 \rfloor$$

Is it true that $0 \leq \hat{q} - \mathrm{div}(a, b) \leq 1$?

$-2^{-8} < \hat{q}_0 - a/b < 2^{-8}$ so $0 < \hat{q}_1 - a/b < 2^{-7} < 1$

3.9 Fixed computation

$$\hat{a} \leftarrow \mathrm{RN}(a)$$
$$\hat{b} \leftarrow \mathrm{RN}(b)$$
$$\hat{q}_0 \leftarrow \mathrm{RN}\left(\hat{a}/\hat{b}\right)$$
$$\hat{q}_1 \leftarrow \mathrm{RN}(\hat{q}_0 + 2^{-8})$$
$$\hat{q} = (\hat{q}_h, \hat{q}_\ell) \leftarrow \lfloor \hat{q}_1 \rfloor$$

Is it true that $0 \leq \hat{q} - \mathrm{div}(a, b) \leq 1$?

$$-2^{-8} < \hat{q}_0 - a/b < 2^{-8} \text{ so } 0 < \hat{q}_1 - a/b < 2^{-7} < 1$$

$$0 \leq \hat{q} - \mathrm{div}(a, b) = \lfloor \hat{q}_1 \rfloor - \lfloor a/b \rfloor \leq 1 \quad ✅$$

3.9 Fixed computation

$$\hat{a} \leftarrow \mathrm{RN}(a)$$
$$\hat{b} \leftarrow \mathrm{RN}(b)$$
$$\hat{q}_0 \leftarrow \mathrm{RN}\left(\hat{a}/\hat{b}\right)$$
$$\hat{q}_1 \leftarrow \mathrm{RN}(\hat{q}_0 + 2^{-8})$$
$$\hat{q} = (\hat{q}_h, \hat{q}_\ell) \leftarrow \lfloor \hat{q}_1 \rfloor$$

Is it true that $0 \leq \hat{q} - \mathrm{div}(a, b) \leq 1$?

$-2^{-8} < \hat{q}_0 - a/b < 2^{-8}$ so $0 < \hat{q}_1 - a/b < 2^{-7} < 1$

$0 \leq \hat{q} - \mathrm{div}(a, b) = \lfloor \hat{q}_1 \rfloor - \lfloor a/b \rfloor \leq 1$ ✅

Since $b \leq 2^{63}$, we have $-2^{63} \leq a - b \cdot \hat{q} < 2^{63}$, which (barely) fits in a signed 64-bit integer. Hence $\mathrm{S}_{64}(\hat{r}) = a - b \cdot \hat{q}$.

4. Divisions by constants

4.1 Unsigned division by a constant, $0 < b < 2^{21}$?

Div64-Case1

Input: $a = (a_h, a_\ell)$, $b = (b_h, b_\ell)$, $\hat{m} \in \mathbb{F}_{64}$, $0 < b < 2^{18}$

Output: $q = (q_h, q_\ell) = \mathrm{div}(a, b)$, $r = (r_h, r_\ell) = \mathrm{rem}(a, b)$

Assert: $b_h = 0$

$q_h \leftarrow \mathrm{div}(a_h, b_\ell)$

$k \leftarrow a_h -_{32} (b_\ell \times_{32} q_h)$

$a' \leftarrow (k, a_\ell) = 2^{32}k + a_\ell$

$q_0 \leftarrow \mathrm{RN}(\mathrm{RN}(a') / \mathrm{FP}(b_\ell))$

Assert: $0 \leq q_0 < 2^{32}$

$q_\ell \leftarrow \lfloor q_0 \rfloor$

$r_h \leftarrow 0$

$r_\ell \leftarrow a_\ell -_{32} (b_\ell \times_{32} q_\ell)$

4.2 Unsigned division by a constant, $0 < b < 2^{21}$?

$$q_0 \leftarrow \mathrm{RN}(\mathrm{RN}(a') / \mathrm{FP}(b_\ell))$$
$$q_\ell \leftarrow \lfloor q_0 \rfloor$$

Requirement: $q_\ell = \mathrm{div}(a', b_\ell) = \lfloor a'/b_\ell \rfloor$

1

1

4.2 Unsigned division by a constant, $0 < b < 2^{21}$?

$$q_0 \leftarrow \mathrm{RN}(\mathrm{RN}(a') \cdot \hat{m})$$
$$q_\ell \leftarrow \lfloor q_0 \rfloor$$

$\hat{m} =$

Requirement: $q_\ell = \mathrm{div}(a', b_\ell) = \lfloor a'/b_\ell \rfloor$

1

1

4.2 Unsigned division by a constant, $0 < b < 2^{21}$?

$$q_0 \leftarrow \mathrm{RN}(\mathrm{RN}(a') \cdot \hat{m})$$
$$q_\ell \leftarrow \lfloor q_0 \rfloor$$

$$\hat{m} = \mathrm{RN}\left(\frac{1 + 2^{-51}}{\mathrm{FP}(b)}\right)$$

Requirement: $q_\ell = \mathrm{div}(a', b_\ell) = \lfloor a'/b_\ell \rfloor$

From[1], if x and y are n-bit integers, and we have a floating-point system with $F \geq n + 3$ bits of mantissa ($F = 53$ for binary64), then

$$\lfloor x/y \rfloor = \lfloor \mathrm{RN}(x \cdot \hat{m}) \rfloor \quad \text{given} \quad \hat{m} = \mathrm{RN}\left(\frac{1 + 2^{2-F}}{y}\right)$$

That requires $a' < 2^{50}$, which we only get when $b < 2^{18}$.

[1] Division by invariant integers using multiplication“ by T. Granlund and P. L. Montgomery, Section 7

4.3 Unsigned division by a constant, $2^{14} < b \leq 2^{63}$

Div64-Case2

Input: $a = (a_h, a_\ell)$, $b = (b_h, b_\ell)$, $\hat{m} \in \mathbb{F}_{64}$, $2^{21} \leq b \leq 2^{63}$

Output: $q = (q_h, q_\ell) = \mathrm{div}(a, b)$, $r = (r_h, r_\ell) = \mathrm{rem}(a, b)$

$\hat{q}_1 \leftarrow \mathrm{RN}(\mathrm{RN}(\mathrm{RN}(a)/\,\mathrm{RN}(b)) + 2^{-8})$

Assert: $0 \leq \hat{q}_1 < 2^{64}$

$\hat{q} = (\hat{q}_h, \hat{q}_\ell) \leftarrow \lfloor \hat{q}_1 \rfloor$

$\hat{r} = (\hat{r}_h, \hat{r}_\ell) \leftarrow a -_{64} (b \times_{64} \hat{q})$

if $\mathrm{S}_{64}(\hat{r}) < 0$ **then**

  $q \leftarrow \hat{q} -_{64} 1 \;\; ; \;\; r \leftarrow \hat{r} +_{64} b$

else

  $q \leftarrow \hat{q} \;\; ; \;\; r \leftarrow \hat{r}$

end

4.4 Unsigned division by a constant, $2^{14} < b \leq 2^{63}$

$$\hat{q}_1 \leftarrow \mathrm{RN}(\mathrm{RN}(\mathrm{RN}(a)/\,\mathrm{RN}(b)) + 2^{-8})$$
$$\hat{q} = (\hat{q}_h, \hat{q}_\ell) \leftarrow \lfloor \hat{q}_1 \rfloor$$

Requirement: $0 \leq \hat{q}_1 - a/b \leq 1$ (implies $0 \leq \hat{q} - \mathrm{div}(a, b) \leq 1$)

4.4 Unsigned division by a constant, $2^{14} < b \leq 2^{63}$

$$\hat{q}_1 \leftarrow \mathrm{RN}(\mathrm{RN}(a) \cdot \hat{m})$$
$$\hat{q} = (\hat{q}_h, \hat{q}_\ell) \leftarrow \lfloor \hat{q}_1 \rfloor$$

Requirement: $0 \leq \hat{q}_1 - a/b \leq 1$ (implies $0 \leq \hat{q} - \mathrm{div}(a, b) \leq 1$)

$\hat{m} =$

4.4 Unsigned division by a constant, $2^{14} < b \leq 2^{63}$

$$\hat{q}_1 \leftarrow \mathrm{RN}(\mathrm{RN}(a) \cdot \hat{m})$$
$$\hat{q} = (\hat{q}_h, \hat{q}_\ell) \leftarrow \lfloor \hat{q}_1 \rfloor$$

Requirement: $0 \leq \hat{q}_1 - a/b \leq 1$ (implies $0 \leq \hat{q} - \mathrm{div}(a, b) \leq 1$)

$\hat{m} = \mathrm{RN}(m)$ given $k \in \mathbb{N}, k \geq 14$ such that $m = 1/b + 2^{-51-k} \leq 2^{-k}$

4.4 Unsigned division by a constant, $2^{14} < b \leq 2^{63}$

$$\hat{q}_1 \leftarrow \mathrm{RN}(\mathrm{RN}(a) \cdot \hat{m})$$
$$\hat{q} = (\hat{q}_h, \hat{q}_\ell) \leftarrow \lfloor \hat{q}_1 \rfloor$$

$$\left| \frac{\mathrm{RN}(\hat{a} \cdot \hat{m}) - am}{am} \right| < 2^{-51}$$

Requirement: $0 \leq \hat{q}_1 - a/b \leq 1$ (implies $0 \leq \hat{q} - \mathrm{div}(a, b) \leq 1$)

$\hat{m} = \mathrm{RN}(m)$ given $k \in \mathbb{N}, k \geq 14$ such that $m = 1/b + 2^{-51-k} \leq 2^{-k}$

$|\hat{q}_1 - am| < 2^{-51} \cdot am \leq 2^{-51} a \cdot 2^{-k} = 2^{-51-k} a$

4.4 Unsigned division by a constant, $2^{14} < b \leq 2^{63}$

EPFL

$$\hat{q}_1 \leftarrow \mathrm{RN}(\mathrm{RN}(a) \cdot \hat{m})$$
$$\hat{q} = (\hat{q}_h, \hat{q}_\ell) \leftarrow \lfloor \hat{q}_1 \rfloor$$

$$\left| \frac{\mathrm{RN}(\hat{a} \cdot \hat{m}) - am}{am} \right| < 2^{-51}$$

Requirement: $0 \leq \hat{q}_1 - a/b \leq 1$ (implies $0 \leq \hat{q} - \mathrm{div}(a, b) \leq 1$)

$\hat{m} = \mathrm{RN}(m)$ given $k \in \mathbb{N}, k \geq 14$ such that $m = 1/b + 2^{-51-k} \leq 2^{-k}$

$|\hat{q}_1 - am| < 2^{-51} \cdot am \leq 2^{-51} a \cdot 2^{-k} = 2^{-51-k} a$

$$\begin{aligned} -2^{-51-k} a < \hat{q}_1 - am &< 2^{-51-k} a \\ -2^{-51-k} a < \hat{q}_1 - a/b - 2^{-51-k} a &< 2^{-51-k} a \\ 0 < \hat{q}_1 - a/b &< 2 \cdot 2^{-51-k} a \end{aligned}$$

4.4 Unsigned division by a constant, $2^{14} < b \leq 2^{63}$

$$\hat{q}_1 \leftarrow \mathrm{RN}(\mathrm{RN}(a) \cdot \hat{m})$$
$$\hat{q} = (\hat{q}_h, \hat{q}_\ell) \leftarrow \lfloor \hat{q}_1 \rfloor$$

$$\left| \frac{\mathrm{RN}(\hat{a} \cdot \hat{m}) - am}{am} \right| < 2^{-51}$$

Requirement: $0 \leq \hat{q}_1 - a/b \leq 1$ (implies $0 \leq \hat{q} - \mathrm{div}(a, b) \leq 1$)

$\hat{m} = \mathrm{RN}(m)$ given $k \in \mathbb{N}, k \geq 14$ such that $m = 1/b + 2^{-51-k} \leq 2^{-k}$

$|\hat{q}_1 - am| < 2^{-51} \cdot am \leq 2^{-51} a \cdot 2^{-k} = 2^{-51-k} a$

$$\begin{aligned} -2^{-51-k} a < \hat{q}_1 - am &< 2^{-51-k} a \\ -2^{-51-k} a < \hat{q}_1 - a/b - 2^{-51-k} a &< 2^{-51-k} a \\ 0 < \hat{q}_1 - a/b &< 2 \cdot 2^{-51-k} a \end{aligned}$$

Finally, $0 < \hat{q}_1 - a/b < 2^{-50-k} a < 2^{-64} \cdot 2^{64} = 1$ ✅

5. Conclusion

5.1 Conclusion

- Algorithms for 64-bit integer divisions on JavaScript without loops
- Near-constant performance, almost always faster than alternatives
- Combines 32-bit integer arithmetic and 64-bit floating point arithmetic
- Constant divisors: use multiplications by floating-point constants
- (Not presented: conversion to string with a single multiplication)

Possible future work:

- Signed divisions without computing absolute values
- Run-time invariant divisors ($\hat{m} = \mathrm{RN}(m)$ requires arbitrary precision)

6. Appendix

6.1 Signed division

SDiv64

Input: $a = (a_h, a_\ell)$, $b = (b_h, b_\ell)$, $b \neq 0$
Output: $q = (q_h, q_\ell) = \mathrm{sdiv}_{64}(a, b)$, $r = (r_h, r_\ell) = \mathrm{srem}_{64}(a, b)$

$a' \leftarrow |\mathrm{S}_{64}(a)|$
$b' \leftarrow |\mathrm{S}_{64}(b)|$
if $b' < 2^{21}$ **then**
  $(q', r') \leftarrow$ Div64-Case1(a', b')
else
  $(q', r') \leftarrow$ Div64-Case2(a', b')
end
$q \leftarrow$ **if** $\mathrm{S}_{64}(a \frown b) < 0$ **then** $0 -_{64} q'$ **else** q'
$r \leftarrow$ **if** $\mathrm{S}_{64}(a) < 0$ **then** $0 -_{64} r'$ **else** r'

6.2 Unsigned conversion to string

ToString

Input: $x = (x_h, x_\ell)$, $\Delta \in \mathbb{N}$, $2 \leq \Delta \leq 36$
Output: the string representation of x in base Δ

```
if x_h = 0, i.e., x = x_ℓ then
    return JSToString(FP(x_ℓ), Δ)
else if x < 2^53 then
    return JSToString(RN(x), Δ)
else
    return ToStringLarge(x, Δ)
end
```

6.3 Signed conversion to string

SToString

Input: $x = (x_h, x_\ell)$, $\Delta \in \mathbb{N}$, $2 \leq \Delta \leq 36$
Output: the string representation of $\mathrm{S}_{64}(x)$ in base Δ

```
if S64(x) = S32(x_ℓ) then
    return JSToString(FP(S32(x_ℓ)), Δ)
else if −2^53 ≤ x < 2^53 then
    return JSToString(RN(S64(x)), Δ)
else if S64(x) < 0 then
    return "-" + ToStringLarge(0 −64 x, Δ)
else
    return ToStringLarge(x, Δ)
end
```

6.4 Large unsigned conversion to string

ToStringLarge

Input: $x = (x_h, x_\ell)$, $x \geq 2^{30}$, $\Delta \in \mathbb{N}$, $2 \leq \Delta \leq 36$
Output: the string representation of x in base Δ

$(d, w, \hat{m}) \leftarrow \textsc{ToStringTable}[\Delta]$
$q_1 \leftarrow \mathrm{RN}(\mathrm{RN}(x) \cdot \hat{m})$
$q \leftarrow \lfloor q_1 \rfloor$ // as a $\mathbb{F}_{64}$
$\hat{q}_\ell \leftarrow \mathrm{truncate}(\hat{q}) \bmod 2^{32}$ // in JS: `ql = q|0`
$\hat{r} \leftarrow x_\ell -_{32} (d \times_{32} \hat{q}_\ell)$
if $\mathrm{S}_{32}(\hat{r}) < 0$ **then**
  $q \leftarrow \mathrm{RN}(\hat{q} - 1)$; $r \leftarrow \hat{r} +_{32} d$
else
  $q \leftarrow \hat{q}$; $r \leftarrow \hat{r}$
end
$s_q \leftarrow \textsc{JSToString}(q, \Delta)$
$s_r \leftarrow \textsc{JSToString}(\mathrm{FP}(r), \Delta)$`.padStart(`$w$`, "0")`
return $s_q + s_r$

$\mathrm{ToStringTable}[\Delta] = (d, w, \hat{m})$

- $d = \Delta^w$ for some integer w
- $2^{30}/\Delta < d \leq 2^{30}$
- $\hat{m} = \mathrm{RN}(m)$
  for $m = 1/d + 2^{-51-k}$
  and $k = 24$