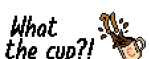
How to get rid of subtraction

What's this 2-complement thing, anyway?

Christophe Deleuze

12/09/2025 

Natural numbers, addition, and subtraction

When we were kids, we learned to perform addition and subtraction on natural numbers

- with efficient algorithms, thanks to the decimal positional numbering system
- $123 = 1x10^2 + 2\times10^1 + 3\times10^0$

```
  123          172
+  49        -  49
-----        -----
  172          123
```

Negative integers, addition, and subtraction

We then extended natural numbers to (signed) integers

- and the "arithmetic negation" operation
- nice property: $a - b = a + (-b)$
  - any subtraction can be expressed as an addition
  - subtraction is superfluous!

Negative integers, addition, and subtraction

We then extended natural numbers to (signed) integers

- and the "arithmetic negation" operation
- nice property: $a - b = a + (-b)$
  - any subtraction can be expressed as an addition
  - subtraction is superfluous!
- but in practice we don't compute this way!
  - asked to compute $43 - 12$, do you perform $43 + (-12)$?

Negative integers, addition, and subtraction

We then extended natural numbers to (signed) integers

- and the "arithmetic negation" operation
- nice property: $a - b = a + (-b)$
  - any subtraction can be expressed as an addition
  - subtraction is superfluous!
- but in practice we don't compute this way!
  - asked to compute $43 - 12$, do you perform $43 + (-12)$?
  - how do you compute this addition: $34 + (-8)$?

That's odd!

- we can't replace subtraction by addition
- we even need the subtraction algorithm for some additions!
- rules to compute $a + b$ in $\mathbb{Z}$

$$\begin{array}{llll} a > 0, & b > 0 & & a + b \\ a > 0 & b < 0, & a \geq |b| & a - |b| \\ a > 0 & b < 0 & a \leq |b| & -(|b| - a) \\ a < 0 & b < 0 & & -(|a| + |b|) \end{array}$$

That's odd!

- we can't replace subtraction by addition
- we even need the subtraction algorithm for some additions!
- rules to compute $a + b$ in $\mathbb{Z}$

$$\begin{array}{llll} a > 0, & b > 0 & & a + b \\ a > 0 & b < 0, & a \geq |b| & a - |b| \\ a > 0 & b < 0 & a \leq |b| & -(|b| - a) \\ a < 0 & b < 0 & & -(|a| + |b|) \end{array}$$

- don't you find this utterly unsatisfying?

That's odd!

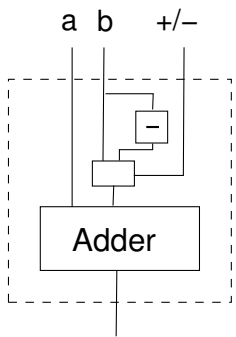
- we can't replace subtraction by addition
- we even need the subtraction algorithm for some additions!
- rules to compute $a + b$ in $\mathbb{Z}$

$$\begin{array}{llll} a > 0, & b > 0 & & a + b \\ a > 0 & b < 0, & a \geq |b| & a - |b| \\ a > 0 & b < 0 & a \leq |b| & -(|b| - a) \\ a < 0 & b < 0 & & -(|a| + |b|) \end{array}$$

- don't you find this utterly unsatisfying?
  - well, maybe not that much but still...

Do computers manage to do that?

Schematic view of an add/subtract circuit



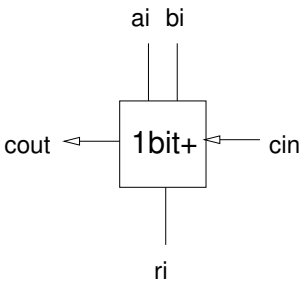
Computer arithmetic is a bit different...

- first difference: use binary positional system
- $1010 = 1\times2^3 + 0\times2^2 + 1\times2^1 + 0\times2^0$

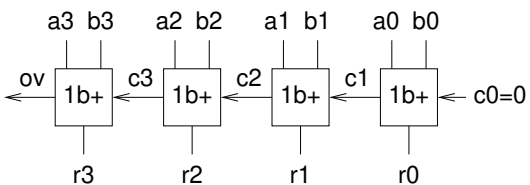
```
 1010
+  10
-----
 1100
```

Performing a binary addition in hardware

one bit adder



4 bit adder



What about subtraction? The 2-complement

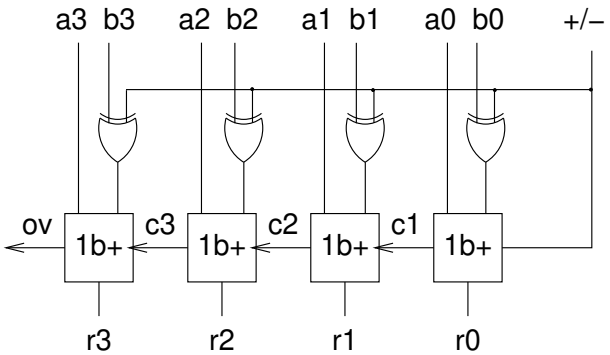
- the [-] box computes the 2-complement
- sounds like the opposite
- but computers manipulate two kinds of integers:
  - signed ints ~ integers
  - unsigned ints ~ natural numbers
- let's consider unsigned ints
- what's the other way computer arithmetic differs?

What about subtraction? The 2-complement

- the [-] box computes the 2-complement
- sounds like the opposite
- but computers manipulate two kinds of integers:
  - signed ints ~ integers
  - unsigned ints ~ natural numbers
- let's consider unsigned ints
- what's the other way computer arithmetic differs?
  - finite arithmetic (eg 4 bit numbers)
- (very) odd property of finite binary arithmetic: $\forall a, a + 2^n = a$
- $a - b = a - b + 2^n = a + (2^n - b)$

Wait, isn't $2^n - b$ a subtraction?

Here's the add/sub circuit in a 4-bit computer



- add: $a_3a_2a_1a_0 + b_3b_2b_1b_0 + 0$
- sub: $a_3a_2a_1a_0 + \overline{b_3b_2b_1b_0} + 1$

Computing the 2-complement

what we want	what the circuit does
$a + (2^n - b)$	$a + (\overline{b} + 1)$

- $a + (2^n - b) = a + (2^n - 1 - b + 1)$
- $2^n - 1 - b$ is the 1-complement of b

```
2-c                    1-c
   10000   (2^4)           1111   (2^4-1)
 - bbbb                 - bbbb
 ------                 ------
```

Computing the 1-c does not involve the subtraction algorithm (no borrow can occur)

Can we do that in the decimal system?

- why couldn't we?

Can we do that in the decimal system?

- why couldn't we?
- $a - b = a + 10^n - b$
- let's compute $382 - 247$
- n=3
- 10-complement of b: $1000 - b$

```
  1000
-  247
------
   753
```

Can we do that in the decimal system?

- why couldn't we?
- $a - b = a + 10^n - b$
- let's compute $382 - 247$
- n=3
- 10-complement of b: $1000 - b$

```
  1000
-  247
------
  753
```

- 9-complement of b: $999 - b$

```
  999
-  247
-----
  752
```

Can we do that in the decimal system?

- why couldn't we?
- $a - b = a + 10^n - b$
- let's compute $382 - 247$
- n=3
- 10-complement of b: $1000 - b$

```
  1000
-  247
------
  753
```

- 9-complement of b: $999 - b$

```
  999
-  247
-----
  752
```

- finally

```
  382
+ 753
-----
 .135
```

Summary

to compute $a - b$

1. choose n
2. compute 9-c of b
3. add 1 to get 10-c of b
4. add a to 10-c of b (throwing away excess digit)

Summary

to compute $a - b$

1. choose n
2. compute 9-c of b
3. add 1 to get 10-c of b
4. add a to 10-c of b (throwing away excess digit)

algorithms involved

- addition
- 9-complement computation (not a subtraction)

Summary

to compute $a - b$

1. choose n
2. compute 9-c of b
3. add 1 to get 10-c of b
4. add a to 10-c of b (throwing away excess digit)

algorithms involved

- addition
- 9-complement computation (not a subtraction)

