Binary Arithmetic

CS 2130: Computer Systems and Organization 1

August 31, 2022

Announcements

- Quiz 0 due Friday at 5pm (when Quiz 1 opens)
- TA office hours start tonight!
  - **In-person**: Olsson 001, Wed-Sun, 5-7pm
  - **Online**: Discord, Wed-Sun, varies
  - Office hour page has been updated
- My office hours start Thursday!
  - Tuesday, 4-5pm, Discord/Zoom
  - Wednesday, 4:30-6pm, Rice 210 (masks requested)
  - Thursday, 11am-12pm, Discord/Zoom
- Lab 1 late check-off through Monday
- Covid-19 make-up policies: stay home, check-off lab later

Numbers

From our oldest cultures, how do we mark numbers?

- Arabic numerals
  - Positional numbering system

Numbers

From our oldest cultures, how do we mark numbers?

- Arabic numerals
  - Positional numbering system
  - The **10** is significant:
    - 10 symbols, using 10 as base of exponent

2130

Numbers

From our oldest cultures, how do we mark numbers?

- Arabic numerals
  - Positional numbering system
  - The `10` is significant:
    - 10 symbols, using 10 as base of exponent
  - The `10` is *arbitrary*
  - We can use other bases! $\pi, 2130, 2, \ldots$

Base-8 Example

Try to turn 134_8 into base-10:

Bases

We will discuss a few in this class

- Base-10 (decimal) - talking to humans
- Base-8 (octal) - shows up occasionally
- Base-2 (binary) - most important! (we've been discussing 2 things!)
- Base-16 (hexadecimal) - nice grouping of bits

Binary

2 digits: 0, 1

$2^6\ 2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0$

Try to turn 1100101_2 into base-10:

64 32 16 8 4 2 1

$64 + 32 + 4 + 1 = 101$

Binary

Any downsides to binary?

Turn 2130_{10} into base-2:
hint: find largest power of 2 and subtract

100001010010

2130
- 2048 2^{11}
0082

82
- 64
18
- 16
2
- 2
0

Long Numbers

How do we deal with numbers too long to read?

Long Numbers

How do we deal with numbers too long to read?

- Group them by 3 (right to left)

Long Numbers

How do we deal with numbers too long to read?

- Group them by 3 (right to left)
- In decimal, use commas: **,**
- Numbers between commas: 000 - 999

Long Numbers

How do we deal with numbers too long to read?

- Group them by 3 (right to left)
- In decimal, use commas: **,**
- Numbers between commas: 000 - 999
- Effectively base-1000

Long Numbers in Binary

Making binary more readable

- Typical to group by 3 or 4 bits
- No need for commas *Why?*

$2^3 = 8$

0 ... 7

100001010010

Long Numbers in Binary

Making binary more readable

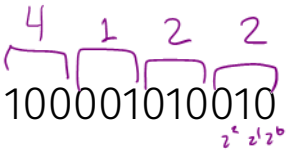
- Typical to group by 3 or 4 bits
- No need for commas *Why?*
- We can use a separate symbol per group
- How many do we need for groups of 3?

100001010010

Long Numbers in Binary

Making binary more readable

- Typical to group by 3 or 4 bits
- No need for commas *Why?*
- We can use a separate symbol per group
- How many do we need for groups of 3? 8
- Turn each group into decimal representation



Long Numbers in Binary

Making binary more readable

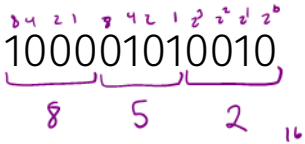
- Typical to group by 3 or 4 bits
- No need for commas *Why?*
- We can use a separate symbol per group
- How many do we need for groups of 3?
- Turn each group into decimal representation
- Converts binary to **octal**

100001010010 4122_8

Long Numbers in Binary

Making binary more readable

- Groups of 4 more common
- How many symbols do we need for groups of 4?



$2^4 = 16$

Long Numbers in Binary

Making binary more readable

- Groups of 4 more common
- How many symbols do we need for groups of 4?
- Converts binary to **hexadecimal**
- Base-16 is very common in computing

100001010010

Hexadecimal

Need more than 10 digits. What next?

8 4 2 1
1110 = e16

0
.
1
1
.

9

a - 10
b - 11
c - 12
d - 13
e - 14
f - 15

Hexadecimal Exercise

Consider the following hexadecimal number:

852dab1e

Is it even or odd?

Using Different Bases in Code

	Old Languages	New Languages
binary	no way	0b 100101
octal	0235	0o 235
decimal	2130	2130
hexadecimal	0x43a	0x43a

Negative Integers

Representing negative integers

Negative Integers

Representing negative integers

- Can we use the minus sign?

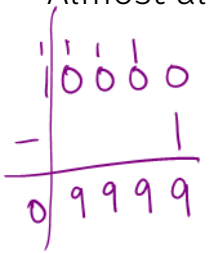
-256

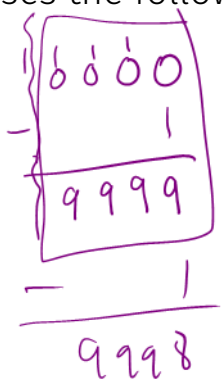
-101011

Negative Integers

Representing negative integers

- Can we use the minus sign?
- In binary we only have 2 symbols, must do something else!
- Almost all hardware uses the following observation:





Negative Integers

Representing negative integers

- Computers store numbers in fixed number of wires
- Ex: consider 4-digit decimal numbers

Negative Integers

Representing negative integers

- Computers store numbers in fixed number of wires
- Ex: consider 4-digit decimal numbers
- Throw away the last borrow:
  - 0000 - 0001 = 9999
  - 9999 - 0001 = 9998
  - Normal subtraction/addition still works

Negative Integers

Representing negative integers

- Computers store numbers in fixed number of wires
- Ex: consider 4-digit decimal numbers
- Throw away the last borrow:
  - 0000 - 0001 = 9999
  - 9999 - 0001 = 9998
  - Normal subtraction/addition still works
- This works the same in binary

Two's Complement

This scheme is called **Two's Complement**

- More generically, a *signed* integer
- There is a break as far away from 0 as possible
- First bit acts vaguely like a minus sign
- Works as long as we do not pass number too large to represent