CS 47

Introduction to Computer Systems

Thomas Howell January 27, 2014

Topics:

Arithmetic

-1- class00.ppt

CS 47 Spring, 2014

Arithmetic

- We need to know how to do arithmetic by hand in order to understand how it is done by the computer.
- Our ordinary arithmetic is done using place-value notation with base 10. Each place is "worth" 10 times the place to its right.
- Computer arithmetic is done using place value notation with base 2. It is often convenient to use a power of 2 for the base, most commonly 2⁴ = 16.
- In base 16, we need numerals (digits) to represent 10, 11, 12, 13, 14, and 15. We use a, b, c, d, e, f (or A, B, C, D, E, F).
- Most of us learned in grade school how to add, subtract, multiply, and divide decimal integers of any length.
- The same algorithms work for any base: 10, 2, 16, or whatever you like.

Integer Arithmetic

Integers

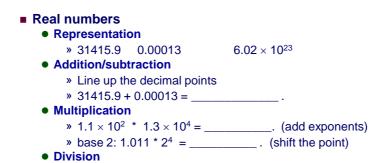
Representation

- » $1728_{10} = 1 * 10^3 + 7 * 10^2 + 2 * 10^1 + 8$
 - » $1011_2 = 1 * 2^3 + 0 * 2^2 + 1 * 2^1 + 1$
- » $3e9_{16} = 3 * 16^2 + (14) * 16^1 + 9$ (digits a-f = 10 15)
- Addition/subtraction (use "carry" and "borrow")
 - » base 10: 1728 + 273 = ____
 - » base 2: 1011 + 101 = ____
- ____ 1728 831 = ____ ___ 1011 – 100 = ____
 - 3e9 1a = ____
- » base 16: 3e9 + 16 = _____
 Multiplication ("digit-by-digit" table)
 - » base 10: 144 * 12 = _____
 - » base 2: 1010 * 101 = ____
 - » base 16: 63 * 7 =
- Division ("long division")
 - » base 10: 1729 / 12 = ____ r. ____
 - » base 2: 1011 / 101 = ____ r. ____
 - » base 16: 3e9 / 7 = ____ r. ____

- 3 -

CS 47 Spring, 2014

Real Arithmetic



» 1.43×10^6 / 1.1×10^2 = _____. (subtract exponents)

Why base-2 and base-16?

- base 2 (binary)
 - Computers are made out of logic circuits. Each logic value is true or false, represented by 1 or 0. Having only two possible values maximizes speed and reliability. Base 2 is the natural way to represent these values.
- base 16 (hexadecimal)
 - Base 2 is cumbersome for human use. There are too many digits. Each base-16 digit represents four base-2 digits.
 - Example: $1111000100110101_2 \rightarrow 1111\ 0001\ 0011\ 0101 \rightarrow f135_{16}$

CS 47 Spring, 2014

Conversion table

If you don't know it already, please learn this table. It will make your life in CS47 (and computer science in general) much easier.

Base 2	Base 10	Base 16
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	10	а
1011	11	b
1100	12	С
1101	13	d
1110	14	е
1111	15	f

-6-

-5-

Converting to a different base • Decimal to hexadecimal (computing in decimal) • $315_{10} = 19 * 16 + b$ $19_{10} = 1 * 16 + 3$ $1_{10} = 0 * 16 + 1$ $315_{10} = 13b_{16}$ OR • $315_{10} = (((3 * 10) + 1) * 10) + 5$ compute in hex: 3 * a + 1 = 1f. 1f * a + 5 = 13b

-7-

CS 47 Spring, 2014

Converting to a different base (2) • Hexadecimal to Decimal • $21d_{16} = 36_{16} * a_{16} + 1$ (21d / a) = 36 r. 1 $36_{16} = 5_{16} * a_{16} + 4$ (36 / a) = 5 r. 4 $5_{16} = 0_{16} * a_{16} + 5$ $311_{16} = 541_{10}$ OR • $21d_{16} = (((2 * 16) + 1) * 16) + d$ compute in decimal: 2 * 16 + 1 = 33. 33 * 16 + 13 = 541

Converting to a different base (3)

- Hexadecimal to Binary
 - 21d₁₆ = 0010 0001 1101
- Binary to Hexadecimal
 - 0101 1011 1001 = 5b9₁₆
- These conversions are effortless when you know the table.
- Writing extra space after every fourth binary digit is helpful.
- Hexadecimal can be regarded as just a more compact way to write binary numbers.

-9-

CS 47 Spring, 2014