CSCI 210: Computer Architecture
Lecture 6: Number Systems

Stephen Checkoway

Oberlin College

Slides from Cynthia Taylor

Announcements

- Problem Set 1 due today 11:59 p.m.

Plan for today

- Learn about positional number systems
  - Base 2 (binary)
  - Base 16 (hexadecimal)
  - Base 8 (octal)
- Learn how to convert between them

- NUMBERS USED IN COMPUTERS ARE ALWAYS IN BINARY

Why we need to learn binary (and other number systems)

- Fundamental to how your computer works
  - Will need a good grasp of binary to understand things like logical operations
  - Will need it a lot when we get to logic gates and how the CPU works
  - Will need to translate to binary to work out examples

- Need to understand it to understand many things like network protocols (IP addresses), bit masking, etc.

Positional Notation

- The meaning of a digit depends on its **position** in a number.
- A number, written as the sequence of digits $d_n d_{n-1} \ldots d_2 d_1 d_0$ in **base** b represents the value

$$d_n b^n + d_{n-1} b^{n-1} + \cdots + d_0 b^0$$

Consider 101

- In base 10, it represents the number 101 (one hundred one)

- In base 2, 101_2 =

- In base 8, 101_8 =

$$101_5 = ?$$

A. 26

B. 51

C. 126

D. 130

$122_{-3}=?$

A. 17

B. 5

C. 10

D. -30

CS History: Negabinary

- Early Polish computers the BINEG (1959) and UMC-1 (1962) used negative binary (base -2)
- Allowed for a natural representation of both negative and positive numbers
- Problem: Math is more complicated

Binary: Base 2

- Used by computers

- A number, written as the sequence of digits $d_n d_{n-1} \ldots d_2 d_1 d_0$ where d is in {0, 1}, represents the value

$$d_n 2^n + d_{n-1} 2^{n-1} + \cdots + d_0 2^0$$

Binary to Decimal

- We have b = 2

$10110_2 =$

Decimal to Binary

- Convert 115 to binary
- We know
  $115 = d_n \cdot 2^n + \cdots + d_1 \cdot 2^1 + d_0 \cdot 2^0$
  $\quad = 2(d_n \cdot 2^{n-1} + \cdots + d_1) + d_0$
- 115 is odd and $2(d_n \cdot 2^{n-1} + \cdots + d_1)$ is even so $d_0 = 1$
- Subtract 1, divide by 2, and repeat
  $57 = d_n \cdot 2^{n-1} + \cdots + d_2 \cdot 2^1 + d_1$
  $\quad = 2(d_n \cdot 2^{n-2} + \cdots + d_2) + d_1$

Convert 115 to Binary

Decimal to Binary

- Repeatedly divide by 2, recording the remainders until you reach 0
- The remainders form the binary digits of the number from the least significant to the most significant
- Converting 25 to binary

$34_{10}=?_2$

A. 010001

B. 010010

C. 100010

D. 111110

E. None of the above

Hexadecimal: Base 16

- Like binary, but shorter!
- Each digit is a “nibble”, or half a byte (4 bits)
- Indicated by prefacing number with 0x (usually)

- A number, written as the sequence of digits $d_n d_{n-1} \ldots d_2 d_1 d_0$ where d is in {0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F}, represents the value

$$d_n 16^n + d_{n-1} 16^{n-1} + \cdots + d_0 16^0$$

Hexadecimal to binary

- Each hex digit corresponds directly to four binary digits

- $35AE_{16}$ =

In MIPS, we can load a hexadecimal constant into a register as normal, e.g.,

```
li   $t0, 0x7E26
```

After this instruction, what value does $t0 hold?

$7E26_{16} = 32294_{10}$

A. 32294 stored in hexadecimal
B. 32294 stored in binary
C. 0x7E26 stored in decimal
D. 0x7E26 stored in binary
E. More than one of the above (which?)

$$23C_{16} = ?_2$$

A. 0010 0000 1100

B. 0010 1111 0010

C. 0010 0011 1100

D. 1000 1101 1000

E. None of the above

Octal: Base 8

- Sometimes used to shorten binary
  - Used to specify UNIX permissions (remember CS 241?)

- A number, written as the sequence of digits $d_n d_{n-1} \ldots d_2 d_1 d_0$ where d is in {0,1,2,3,4,5,6,7}, represents the value

$$d_n 8^n + d_{n-1} 8^{n-1} + \cdots + d_0 8^0$$

$$31_8 = ?_{10}$$

A. 24

B. 25

C. 200

D. 208

E. None of the above

If every hex digit corresponds to 4 binary digits, how many binary digits does an octal digit correspond to?

A. 2
B. 3
C. 4
D. 5

Addition

- Use the same place-by-place algorithm that you use for decimal numbers, but do the arithmetic in the appropriate base

$$\begin{array}{r} \text{2A5C} \\ \underline{+\ \text{38BE}} \end{array}$$

A. 586A

B. 631A

C. 6986

D. None of the above

Reading

- Next lecture: Negatives in binary
  - Reading: rest of section 2.4

- Problem Set 1 due today