CSE 30: Computer Organization and Systems Programming

Lecture 2: Number Representation

Diba Mirza
University of California, San Diego
Levels of Representation

High Level Language Program (e.g., C)

Assembly Language Program (e.g., ARM)

Machine Language Program (ARM)

Compiler

Assembler

Machine Interpretation

Hardware Architecture Description (e.g., block diagrams)

Architecture Implementation

Logic Circuit Description (Circuit Schematic Diagrams)

temp = v[k];
v[k] = v[k+1];
v[k+1] = temp;

ldr r0, [r2]
ldr r1, [r2, #4]
str r1, [r2]
str r0, [r2, #4]

0000 1001 1100 0110 1010 1111 0101 1000
1010 1111 0101 1000 0000 1001 1100 0110
1100 0110 1010 1111 0101 1000 0000 1001
0101 1000 0000 1001 1100 0110 1010 1111
Recall the stored program model

Q: A key feature of the stored program model is that computation is performed
A. by an electronic circuit consisting of transistors
B. on instructions and data that reside in memory
C. on instructions that have to necessarily be in binary format
D. None of the above
## Encoding information

- Encoding numbers (unsigned) using *positional encoding*

<table>
<thead>
<tr>
<th>Digits</th>
<th>Base</th>
<th>Example</th>
</tr>
</thead>
</table>

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

Why is each important??
How many bits are needed to represent any hex digit?

A. 6
B. 5
C. 4
D. 3
\[101_5 = ?\]

A. 26

B. 51

C. 126

D. 130
External vs. Internal Representation

• External representation:
  – Convenient for programmer

• Internal representation:
  – Actual representation of data in the computer’s memory and registers: Always binary (1’s and 0’s)

• Important to be able to convert between the two
Base Conversions

• You should be able to convert between any two bases
  – Decimal to Binary
  – Decimal to hex
  – Hex to binary
10110_2 = ? in decimal
Decimal to binary: $34_{10} = ?_2$

A. 10001

B. 10010

C. 10 0010

D. 111110
Hex to binary: $2 \ 3 \ C_{16} = ?_2$

A. 0010 0011 1010

B. 1011 1110

C. 10 0011 1100

D. 1000011011000
Hex to binary

• Each hex digit corresponds directly to four binary digits

• $35AE_{16} = 0011010110101110_2$ ($3 = 0011$, $5 = 0101$, $A = 1010$, $E = 1110$)
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
Hexadecimal to decimal

$25B_{16} = ?$ Decimal
Hexadecimal to decimal

- Use polynomial expansion

- $25B_{16} = 2 \times 256 + 5 \times 16 + 11 \times 1 = 512 + 80 + 11 = 603$
Decimal to hex: $26_{10}=?_{16}$

A. 19

B. 1A

C. 1B
BIG IDEA: Bits can represent anything!!

- Numbers
- Characters
- Logical values
- Colors: Red, Green, Blue
- Memory addresses
- Instructions

N bits can be used to represent at most $2^N$ things.
How We Store Numbers

• Binary numbers in memory are stored using a finite, fixed number of bits typically:
  8 bits (byte)
  16 bits (half word)
  32 bits (word)
  64 bits (double word or quad)
• If positive pad extra digits with leading 0s
• A byte representing $4_{10} = 00000100$
A byte can store unsigned values up to

A. 127
B. 128
C. 255
D. 256

Generalize to N bits
Integer representation

• Think about representation and storage together
• The number of bits available (finite and fixed) determines:
  – the exact representation of numbers in memory
  – the range of values that can be correctly represented
Unsigned Numbers

• Some types of numbers, such as memory addresses, will never be negative.

• Some programming languages reflect this with types such as “unsigned int”, which only hold positive numbers.

• In an unsigned byte, values will range from 0 to 255.
How to Represent Signed Numbers?

• Representation called **sign and magnitude**
  – Have a separate bit for sign (Most significant Bit MSB)
  – Set it to 0 for positive, and 1 for negative

  Wheel of numbers
In sign and magnitude, we will be able to represent ______ in one byte

A. -63 to 63
B. -127 to 127
C. -128 to 127
D. -255 to 255
E. 0 to 255
Number Representation

• What do we do with numbers?
Shortcomings of sign and magnitude?

• Arithmetic circuit complicated
  – Special steps depending whether signs are the same or not

• Also, two zeros
  – $0x00000000 = +0_{ten}$
  – $0x80000000 = -0_{ten}$

• Therefore sign and magnitude abandoned!
One’s Complement

• To make a number negative, just flip all its bits!
What is the range of numbers we can represent in One’s complement format using N bits

A. $-2^{N-1}$ to $2^{N-1}$

B. $-2^{N-1}$ to $2^{N-1} - 1$

C. $-(2^{N-1}-1)$ to $2^{N-1}-1$

D. $-2^N$ to $2^N$
Shortcomings of One’s Complement

- Still have two zeros (11111111 and 00000000)
- Need an extra step in addition if there is a carry out – see example 1.6 in ARM book
Two’s Compliment

- Each two’s compliment number is now
  \[-2^{n-1}d_{n-1} + 2^{n-2}d_{n-2} + \ldots + 2^1d_1 + 2^0d_0\]
Two’s Complement

• Flip all the bits of unsigned representation and add 1
Two’s Complement: $1101_2 = ?_{10}$

A. -2

B. -3

C. -4

D. -5
Negating Two’s’s Compliment

• To negate any number, flip all the bits and add 1
The negation of $11110001_2$ is ______$2$

A. 00001110

B. 00001111

C. 00011110

D. 01110001
The negation of $11110001_2$ is $______2$

A. 00001110

B. 00001111

C. 00011110

D. 01110001
Sign Extension
Addition and Subtraction

- Positive and negative numbers are handled in the same way.
- The carry out from the most significant bit is ignored.
- To perform the subtraction $A - B$, compute $A + (\text{two's complement of } B)$