Computer Organization and Programming

Raymond Bian

January 11, 2024

Contents

1 Big Ideas **1**

2 Data Types **1**

2.1 Two's Complement 1

Lecture 1: Introduction

1 Big Ideas

Definition 1. Big Idea 1: All computers can compute the same kinds of things. We call this **turing-completeness**.

Definition 2. Big Idea 2: **Abstraction** represents the layers that make the electorns work.

Definition 3. Big Idea 3: **Binary numbers** are better than decimal for electronic computing. Saves energy and is easier to distinguish.

Definition 4. Big Idea 4: Computers store **finite-sized representations** of data and information.

2 Data Types

What does the number 5 mean to us? Does it mean the character 5, the integer 5, or the floating point number 5.0? It could also be represented by roman numberals, etc. These are all many different representations of the idea of "5".

Definition 5. A **data representation** is the set of values from which something can take its value. It also includes the meaning of those values

Example. These representations can be integers, floats, instructions, pointers, addresses, etc. All of these types have various flavors depending on the size, etc.

You can usually expect the processor to deal with data representations gracefully.

Definition 6. A **bit** quite literally, is a charge on the capacitor in the DRAM. It is short for "binary digit" and takes on the value of 0 or 1.

Example. How many bits would we need to store 5 different items?

Proof. 3, as $2^2 \leq 5 \leq 2^3$.

Example. How many different numbers can be represented by 7 bits?

Proof. $2^7 = 128$ numbers.

Definition 7. **Unsigned integers** are represented in bits as a base 2 number mathematically.

For binary numbers, addition and subtraction are defined in exactly the same way. However, you have to carry the bits more often.

Lecture 2: Two's Complement

Multiplication in binary can be done like normal multiplication, but it is a lot faster. Note that we don't even need a microprocessor to add numbers. It can be done purely with transistors.

Definition 8. The **not** operator, denoted $\neg$, toggles the value of a variable.

2.1 Two's Complement

How do we represent negative numbers? We could have signed magnitude, where 1 bit is used on the left. We could also use 1's complement, in which the negative number is just the negated bit value of the positive.

Definition 9. **Two's Complement** wraps back around after the maxmimum number to the least negative number. You can find the negative value of a number by negating it and adding 1.

To extend the side of a number, we fill in to the left copies of the leading bit.

Example. Sum of 101111_2 and 001010_2 is 111001_2.

What should we do when we overflow?