

Slide 1

Minute Essay From Last Lecture

- Several people mentioned seeing AND/OR gates in CSCI 1323. Maybe with some instructors? Ideas are the same as and/or operators but usage is somewhat different I think.
- Two people mentioned their use in Minecraft(!).

Integer Multiplication and Division, Recap

 Algorithms for both operations are based on how you do things "by hand", with some modifications to permit simpler hardware. It's not critical to understand the details, but probably useful to work through an example to believe that it works.

Slide 3

- Required hardware is something that can add two 32-bit numbers, a 64-bit "work area", something to do right and left shifts of the 64-bit area, and some control logic.
- MIPS architecture uses "special registers" 10 and hi for the 64-bit work area. This is where the results end up. There are instructions to multiply, to divide, and to move from the special registers. ("Move from" explains the names of the instructions.)

Representing Real (Non-Integer) Numbers

• Approach is based on a binary version of "scientific notation":

In base 10, we can write numbers in the form $+/-x.yyyy \times 10^{z}$. E.g., $428 = 4.28 \times 10^{2}$, or $-.0012 = -1.2 \times 10^{-3}$.

• We can do the same thing in base 2. Examples:

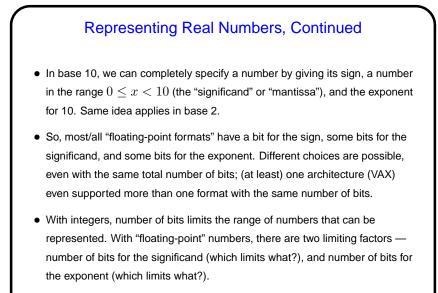
$$32 = 1.0_2 \times 2^5$$
$$-3 = -1.1_2 \times 2^1$$

 $1/2 = 1.0_2 \times 2^{-1}$

$$3/8 = 1.1_2 \times 2^-$$

 This is "floating point" (as opposed to "fixed point", which would allow for non-integers but wouldn't allow as much flexibility — wide range, all with reasonable precision).

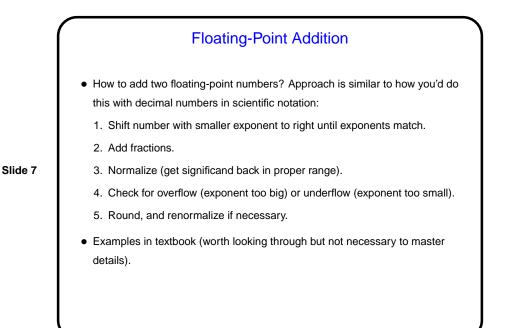
Slide 5

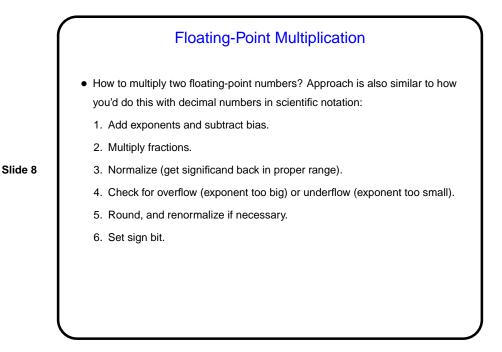


(Does this suggest why the VAX designers offered two formats?)

Representing Real Numbers, Continued

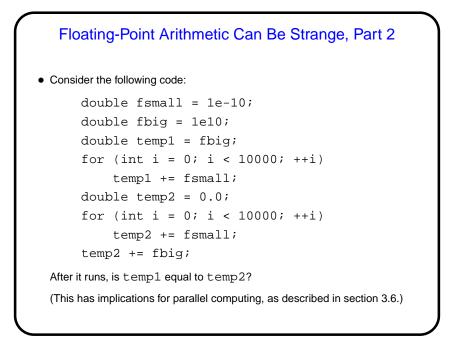
- "IEEE 754 standard" defines formats and operations.
- Exponent is actually stored "biased" actual exponent plus bias (so we only have to store non-negative exponents simplifies comparisons).
- Significand doesn't include leading 1. (Why not?)
- But then how to represent 0? Agree that exponent of all 0s will represent 0 if significand is 0, else "de-normalized number".
- Also, agree that exponent of all 1s will represent +/- "infinity" if significand is 0, else NaN ("not a number" — result of indeterminate or invalid operations such as 0/0).
- "Single-precision" format has 8 bits for exponent, biased by 127, 23 bits for significand. (Double precision is 8, 1023, 52 respectively.)
- Work through an example ...

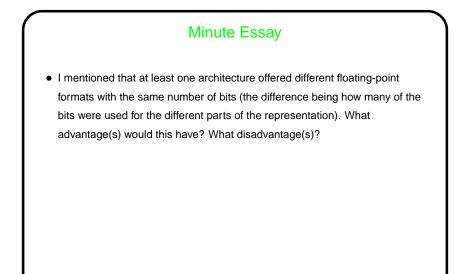




Floating-Point Arithmetic Can Be Strange, Part 1
• Consider the following loop:
 for (f = 0.0; f != 1.0; f += 0.1)
 printf("f = %g\n", f);
What do you think it does?
Why?

Slide 9





Slide 11

Minute Essay Answer

• An advantage is that users would then have a choice between a larger range of possible values and greater precision. A disadvantage is that it's more complicated — both to implement and to use.