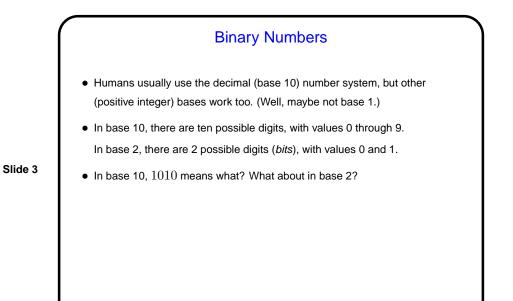
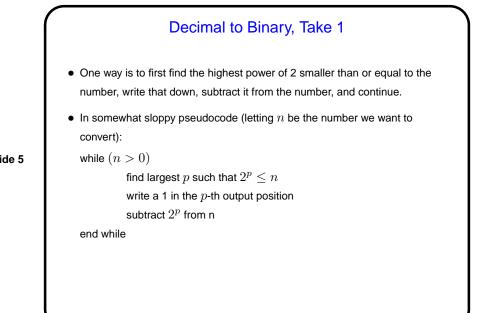


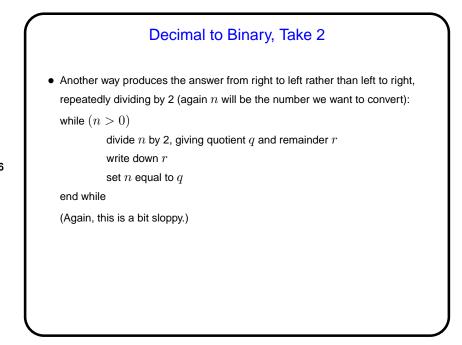
"It's All Ones and Zeros" At the hardware level, all data is represented in binary form — ones and zeros. (Why? hardware for that is simpler to build.) How then do we represent various kinds of data? First a short review of binary numbers ...

Slide 2



Converting Between Bases
Converting from another base to base 10 is easy if tedious (just use definition).
Converting from base 10 to another base? Two algorithms for that ...





Slide 6



 Binary numbers are convenient for computer hardware, but cumbersome for humans to write. Octal (base 8) and hexadecimal (base 16) are more compact, and conversions between these bases and binary are straightforward.

Slide 7

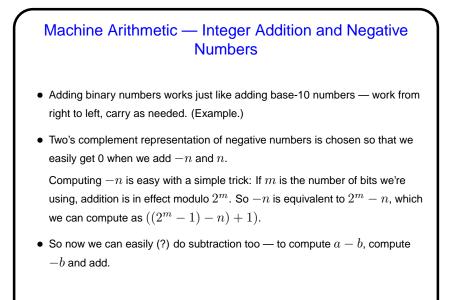
- To convert binary to octal, group bits in groups of three (right to left), and convert each group to one octal digit using the same rules as for converting to decimal (base 10).
- Converting binary to hexadecimal is similar, but with groups of four bits. What to do with values greater than 9? represent using letters A through F (upper or lower case).

Computer Representation of Integers

 So now you can probably guess how non-negative integers can be represented using ones and zeros — number in binary. Fixed size (so we can only represent a limited range).

Slide 8

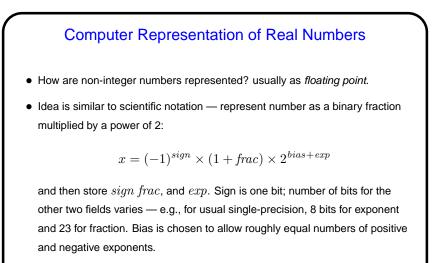
• How about negative numbers, though? No way to directly represent plus/minus. Various schemes are possible. The one most used now is *two's complement*: Motivated by the idea that it would be nice if the way we add numbers doesn't depend on their sign. So first let's talk about addition ...



Binary Fractions
We talked about integer binary numbers. How would we represent fractions?
With base-10 numbers, the digits after the decimal point represent negative powers of 10. Same idea works in binary.

Slide 9

Slide 10



• Current most common format --- "IEEE 754".



- The integers and real numbers of the idealized world of math have some properties not completely shared by their computer representations.
- Math integers can be any size; computer integers can't.
- Math real numbers can be any size and precision; floating-point numbers can't. Also, some quantities that can be represented easily in decimal can't be represented in binary.
- Math operations on integers and reals have properties such as associativity that don't necessarily hold for the computer representations. (Yes, really!)



