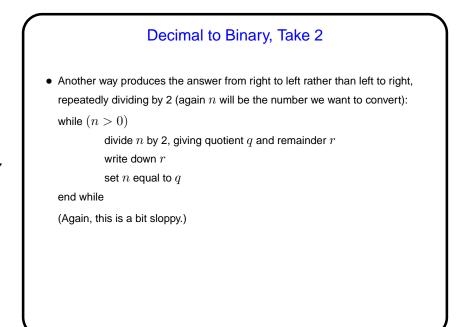
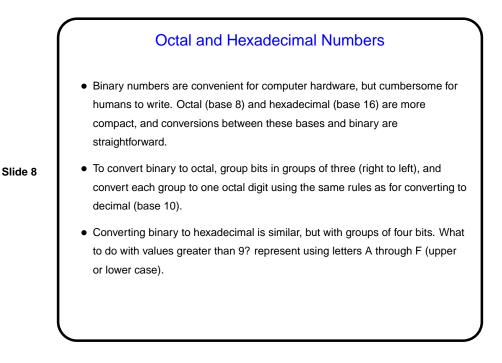


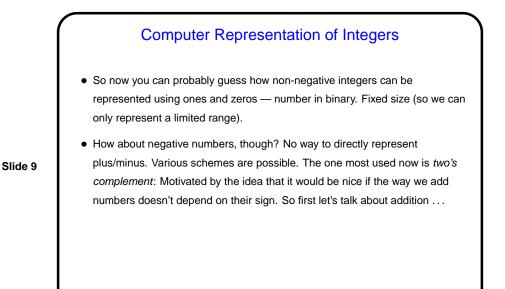
Decimal to Binary, Take 1
One way is to first find the highest power of 2 smaller than or equal to the number, write that down, subtract it from the number, and continue.
In somewhat sloppy pseudocode (letting n be the number we want to convert):

while (n > 0)
find largest p such that 2^p ≤ n
write a 1 in the p-th output position
subtract 2^p from n

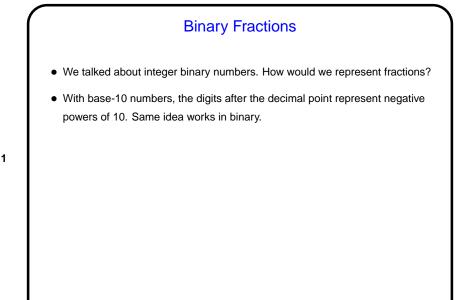
end while







Machine Arithmetic — Integer Addition and Negative **Numbers** • Adding binary numbers works just like adding base-10 numbers — work from right to left, carry as needed. (Example.) • Two's complement representation of negative numbers is chosen so that we easily get 0 when we add -n and n. Computing -n is easy with a simple trick: If m is the number of bits we're using, addition is in effect modulo 2^m . So -n is equivalent to $2^m - n$, which we can compute as $((2^m - 1) - n) + 1)$. • So now we can easily (?) do subtraction too — to compute a - b, compute -b and add.



Computer Representation of Real Numbers

- How are non-integer numbers represented? usually as floating point.
- Idea is similar to scientific notation represent number as a binary fraction multiplied by a power of 2:

Slide 12

$$x = (-1)^{sign} \times (1 + frac) \times 2^{bias + exp}$$

and then store $sign \ frac$, and exp. Sign is one bit; number of bits for the other two fields varies — e.g., for usual single-precision, 8 bits for exponent and 23 for fraction. Bias is chosen to allow roughly equal numbers of positive and negative exponents.

• Current most common format — "IEEE 754".

