Slide 1

Administrivia

- Reminder: Homework 9 due today.
- Reminder: Quiz 6 Wednesday. Likely topic is `structs`.

Slide 2

Bitwise Operators

- In what we've done so far, we've dealt with most data without needing to know exactly how it's represented in terms of 0s and 1s (though knowing a little about that helps you understand limitations and pitfalls).
- However, for various reasons it can be useful or even necessary to work with individual bits — e.g., working with image data (where a "pixel" is represented by some collection of n-bit fields), or working at a very low level with I/O devices. Some system-specific functions callable from C also take as parameters integers that are the result of combining bits.
- So C, like many programming languages, provides operators to allow that . . .

Slide 3

Bitwise Operators and C

- Bitwise "and" (both bits): `&`
- Bitwise "or" (either bit): `|`
- Bitwise "exclusive or" (either bit, not both): `^`
- Bitwise negation/complement (unary operator, flips bits): `~`
- Left and right shifts (specify how many bits): <<, >>).
- All work on integer types.

Slide 4

Bitwise Operators and C, Continued

- In many programming languages, sizes of integer types are fixed, which can make it easier to do this kind of thing.
- In C, however . . . (so you need to be a little careful). (Or if it matters, C99 introduced some fixed-size integer types, in `<stdint.h>`.)

Slide 5

Bitwise Operators and C, Continued

- A typical "use case" for these operators is in working with an integer that's not really an integer so much as a collection of bits, each with a meaning (flags used to communicate with an I/O device, e.g.).
- Typically define "masks" for individual bits or collections of bits, giving them names (via `#define`) and then use bitwise operators to set, clear, test.
- (Example.)

Slide 6

Bitwise Operators and C, Continued

- As another example of using some of these operators and also of using a `union`, we could write a program to show the bits in a `long`, two ways.
- Trying it, on a 64-bit system and also on a rather old 32-bit system, some results are surprising.

Slide 7

Minute Essay

- What is the result of applying bitwise operators as follows:

  1110_2 `&` 1001_2 (bitwise and)

  1110_2 `|` 1001_2 (bitwise or)

  1110_2 `^` 1001_2 (bitwise exclusive or)

  `~`1110_2 (bitwise negation)

- Have you seen anything like this in another course? (What?)
- Anything noteworthy to report about Homework 9? (Yes, I asked last time, but only about half the class was present!)

Slide 8

Minute Essay Answer

- What is the result of applying bitwise operators as follows:

  1110_2 `&` 1001_2 is 1000_2

  1110_2 `|` 1001_2 is 1111_2

  1110_2 `^` 1001_2 is 0111_2

  `~`1110_2 is 0001_2