Bit-fiddling

- Using software to perform logic on individual (or groups) of bits
- The primary way that software controls hardware is by manipulating individual bits in certain hardware **registers** (memory locations)
  - Set a bit to 1
  - Clear a bit to 0
  - Check the value of a bit
- Because computers do not access anything smaller than a byte (8-bits), we must use logic operations to manipulate individual bits within a byte

Numbers in Other Bases in C/C++

- Suppose we want to place the binary value 00111010 into a char v
  - v=58
  - v=0x3a
  - v=0b00111010
- Compilers convert EVERYTHING to equivalent binary

Modifying Individual Bits

- Suppose we want to change only a single bit without changing the other bits
  - v=1? No, assignments changes ALL bits in a variable
- Use bit wise operations
  - AND - clear individual bits to 0
  - OR - set individual bits to 1
  - XOR - invert bits
  - AND - check a bit in a register
- bit = 11110000
- control = 00111100
  - AND - 00110000 (&)
  - OR - 11111100 (|)
  - XOR - 11001100 (^)
  - NOT - 00001111 (~)
- Bitwise operations are used for bit fiddling
- Determine appropriate constant bit patterns (aka masks) that will change some bits while leaving others unchanged
  - Clear LSB to 0 w/o affecting other bits
    - v = v & 0xfe
    - v = v & ~(0x01)
  - Set MSB to 1 w/o affecting other bits
    - v = v | 0x80
  - Flip the LS 4-bits
    - v = v ^ 0x0f