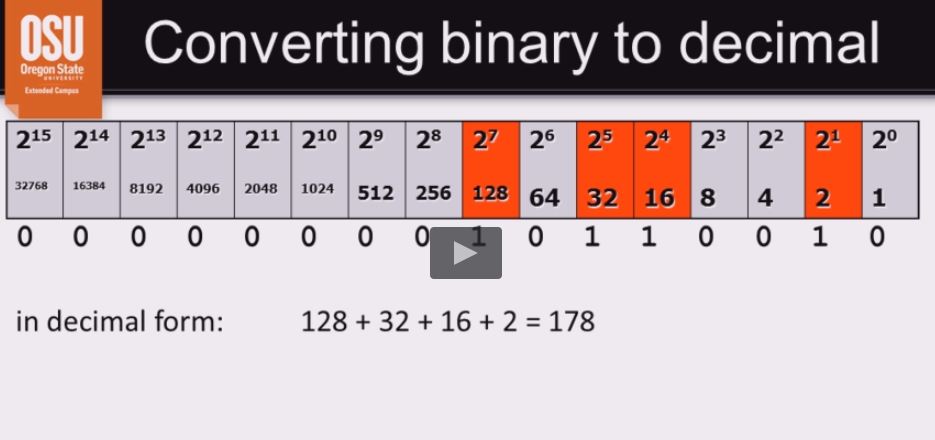
CS271

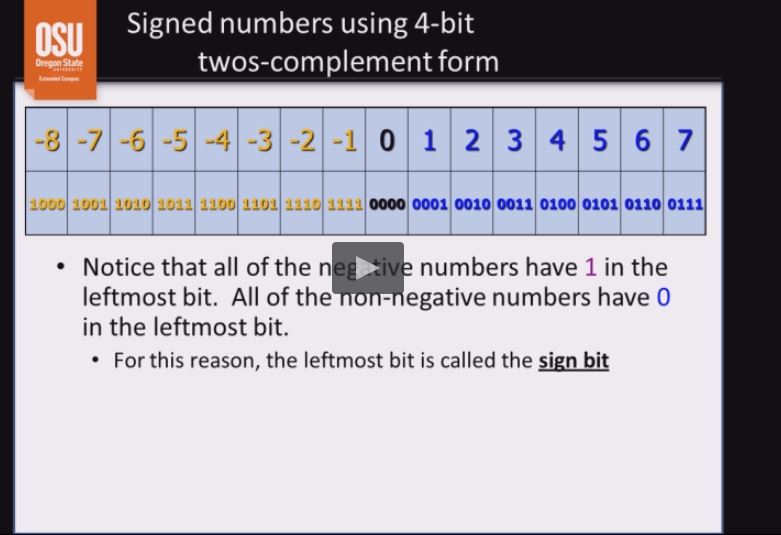
Week3

Data Representation

* Internal rep
  + Switches off and on
  + With open/closed gates
* Two states for each gate
* Binary Number system uses two digit, 0 and 1
* We use standard external representation to transcribe the computer’s internal representation
  + Off is written as 0
  + On is written as 1
* Use binary number sys to rep numeric values electrically
* Switches are grouped into bytes words, etc to represent numerical value
  + Byte = 8 bits, word = 16 bits, etc
* Binary number system
  + Has 2 digits: 0 and 1 (Binary digit)
  + Has places and place values determined by powers of 2
* In theory, any integer value can be represented by this system.
* In a computer, it is finite, representations with too many digits get truncated
* Place values, right to left are 2^0, 2^1, 2^2, etc
* Bits are numbered right to left start at 0
* Place value depends on number of “bits” defined for the type.
* To convert to its familiar decimal rep. just add up the place values of the places that are on



* Example: 157 – converting decimal to binary
  + Method 1: removing largest power of 2
    - 157-128 =29
      * 1 in 128s place
    - 29-16 = 13
      * 0 in 64’s place, 0 in 32’s place, 1 in 16’s place
    - 13-8 = 5
      * 1 in the 8’s place
    - 5-4 = 1
      * 1 in the 4’s place
    - 1-1 = 0
      * 0 in the 2’s place, 1 in the 1’s place
    - RESULT: 10011101
  + Method 2: successive division by 2
    - 157/2 = 78 R 1
    - 78/2 = 39 R 0
    - 39/2 = 19 R 1
    - 19/2 = 9 R 1
    - 9/2 = 4 R 1
    - 4/2 = 2 R 0
    - 2/2 = 1 R 0
    - 1/2 = 0 R 1
    - RESULT
      * 10011101
* External representation
  + Every integer number has a unique representation in each “base” >= 2
  + Hexadecimal is commonly used for easily converting binary to a more manageable form.
    - Because 16 = 2^4, so 4 binary digits can be represented as one hex digit
  + The hexadecimal number system has 16 digits:
    - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E
  + Place values (right to left) are 16^0, 16^1, 16^2, etc..
    - 16^0 = 2^0, 16^1 = 2^4, 16^2 = 2^8, 16^3 = 2^12, etc..
* Hexadecimal
  + Example: 6077(decimal)
    - * 16bit binary <-> hexadecimal
    - Binary 0001 0111 1011 1101
    - Hexadecimal 1 7 B D
      * Write it as 0x17BD or 17BDh
      * 6077/16 = 379 R 13
      * 379/16 = 23 R 11
      * 23/16 = 1 R 7
      * 1/16 = 0 R 1
    - 1 7 (11=B) (13=D) so 17BD
  + Use same method as decimal to binary when converting decimal to hexadecimal
    - 157 decimal = 9D (0x9D or 9Dh)
    - 157 = 10011101
      * 1001 1101
      * 9 13
      * 9 D
        + 0x9D
  + Specify n: number of bits! (8, 16, 32, etc)
    - There are 2^n possible codes
  + Separate the codes so that half of them represente negative numbers
* Twos-complement form
  + For negative number x:
    - Specify number of bits
    - Start with binary represe. Of |x|
    - Change every bit to its opposite, then add 1 to the result
  + EX: -13 in 16bits twos-complement
    - |-13| = 13 = 0000000000001101
    - Ones-complement is 1111111111110010
    - Add 1 to get 1111111111110011 = -13
    - Note that –(-13) should give 13
  + Hexadecimal rep?
    - Convert to hex in the usual way
    - -13 = 1111111111110011 = FFF3 G = 0xfff3
  + Convert negative binary to decimal?
    - Find twos complement, convert, and prepend a minus sign



* N-bit twos-complement form
  + The 2^n possible codes give
    - Zero (all bits are 0)
    - (2^n-1 – 1) positive numbers
    - 2^n-1 negative numbers
  + Note: zero is its own complement
  + Note: there is one (weird) number
    - 01111111 + 1 = 10000000
    - 127 + 1 = -128
      * Inconsistent with rules of arithmetic
    - 127 is the largest number that can be represented in 8bits
    - This means that –(128) cannot be represented with 8 bits
* Signed of Unsigned?
  + A 16-bit rep. could be used for signed or unsigned numbers
    - 16-bit unsigned range is 0…65535
    - 16-bite signed range is -32768 ..+32767
  + Both form use the same 65536 codes
  + Example
    - 1010101010101010 unsigned is 43690 decimal
    - 1010101010101010 signed is -21846 decimal
  + Example
    - 1111111111111111 unsigned is 65535 decimal
    - 1111111111111111 signed is -1 decimal
  + Program must specify
* Negative hex ( signed integers )
  + How can you tell?
    - Recall that a 16-bit signed integer is negative if the let most bit is 1
  + Signed binary Left most digit can be interpreted as a 4-bit binary number
  + If number is 7 or less it will have 0 as left most
  + If 8 or greater it will have a 1 left most bit
  + 16-bit (4 hex digits) examples:
    - 0x7a3e is positive
    - 0x8a3e is negative
    - 0xFFFF is negative
  + Characters
    - ASCII 256 codes, (1-byte)
      * A-Z = 65-90
      * 0-9 = 48-57
    - Unicode 65536 codes (2-byte)
    - Device controllers translate codes (device dependent)
    - All keyboard input is character (including digits)

CS271

Week 4

Binary A