Lecture 3: Bits, bytes, binary numbers, and the representation of information

- **computers represent, process, store, copy, and transmit everything as numbers**
  – hence "digital computer"

- **the numbers can represent anything**
  – not just numbers that you might do arithmetic on

- **the meaning depends on context**
  – as well as what the numbers ultimately represent
  – e.g., numbers coming to your computer or phone from your wi-fi connection could be email, movies, music, documents, apps, Zoom meeting, ...

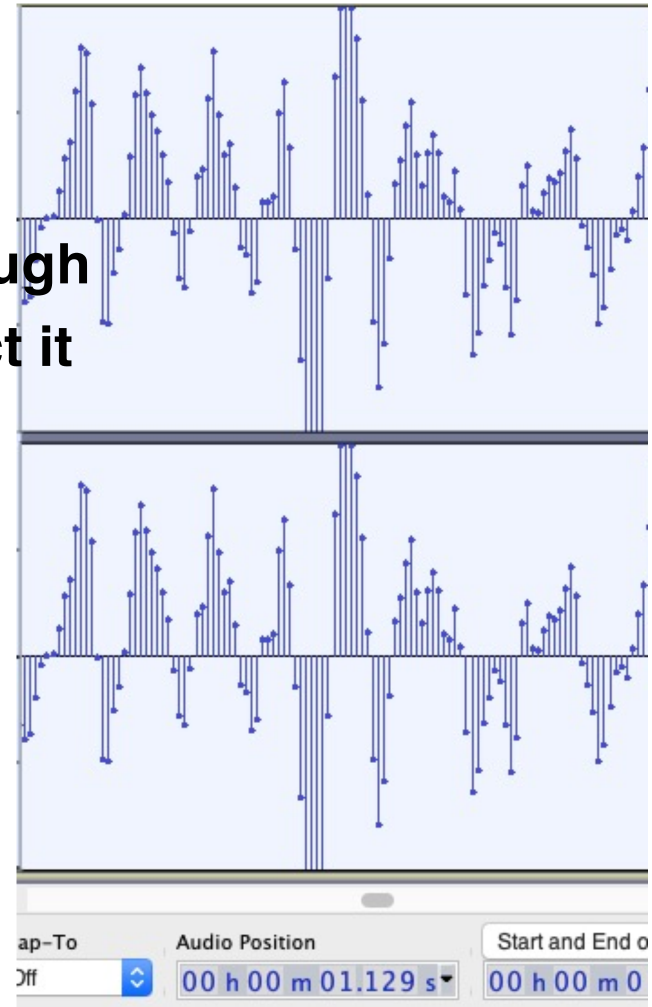
Analog versus Digital

- **analog: "analogous" or "the analog of"**
  - smoothly or continuously varying values
  - volume control, dimmer, faucet, steering wheel
  - value varies smoothly with something else
    - no discrete steps or changes in values
      - small change in one implies small change in another
      - infinite number of possible values
  - the world we perceive is largely analog
- **digital: discrete values**
  - only a finite number of different values
  - a change in something results in sudden change from one discrete value to another
    - digital speedometer, digital watch, push-button radio tuner, …
  - values are represented as <u>numbers</u>

Digital pictures

- divide the picture up into a grid of little rectangles ("pixels")
- assign a different numeric value to each different color value
- the finer the grid and the finer the color distinctions,
  the more accurate the representation will be

Digital sound



- **need to measure intensity/loudness often enough and accurately enough that we can reconstruct it well enough**
- **higher frequency = higher pitch**
- **human ear can hear ~ 20 Hz to 20 KHz**
  - taking samples at twice the highest frequency is good enough (Nyquist)
- **CD audio usually uses**
  - 44,100 samples / second
  - accuracy of 1 in 65,536 (= 2^16) distinct levels
  - two samples at each time for stereo
  - data rate is 44,100 x 2 x 16 bits/sample
    = 1,411,200 bits/sec = 176,400 bytes/sec ~ 10.6 MB/minute
- **MP3 audio compresses by clever encoding and removal of sounds that won't really be heard**
  - data rate is ~ 1 MB/minute

Binary numbers: only use the digits 0 and 1 to represent numbers

- **just like decimal except there are only <u>two</u> digits: 0 and 1**

- **everything is based on powers of 2 (1, 2, 4, 8, 16, 32, …)**
  - instead of powers of 10 (1, 10, 100, 1000, …)

- **counting in binary or base 2:**

  0 1

  1 binary digit represents 1 choice from 2; counts 2 things; 2 distinct values

  00 01 10 11

  2 binary digits represents 1 choice from 4; 4 distinct values

  000 001 010 011 100 101 110 111

  3 binary digits …

- **binary numbers are shorthands for sums of powers of 2**

  11011 = 1 x 16 + 1 x 8 + 0 x 4 + 1 x 2 + 1 x 1

  $= 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$

- **counting in "base 2", using powers of 2**

Binary (base 2) arithmetic

- works like decimal (base 10) arithmetic, but simpler

- addition:

  0 + 0 = 0
  0 + 1 = 1
  1 + 0 = 1
  1 + 1 = 10 ("1 + 1 is 0 and carry the 1")

- subtraction, multiplication, division are analogous

Converting binary to decimal

from right to left:

if bit is 1 add corresponding power of 2

i.e. 2^0, 2^1, 2^2, 2^3

(rightmost power is zero)

$$1101 = 1 \times 2^0 + 0 \times 2^1 + 1 \times 2^2 + 1 \times 2^3$$
$$= 1 \times 1 + 0 \times 2 + 1 \times 4 + 1 \times 8$$
$$= 13$$

Converting decimal to binary

repeat these steps while the number is > 0:

    divide the number by 2

    write the remainder (0 or 1)

    use the quotient as the number and repeat

the answer is the resulting sequence

    in reverse (right to left) order

divide 13 by 2, write "1", number is 6

divide 6 by 2, write "0", number is 3

divide 3 by 2, write "1", number is 1

divide 1 by 2, write "1", number is 0

answer is 1101

Using bits to represent information

- **AB / BSE**
  - 1 bit
- **Fr / So / Jr / Sr**
  - 2 bits
- **grads, auditors, faculty as well**
  - 3 bits
- **a unique number for each person in COS 109**
  - 5 bits
- **a unique number for each freshman at PU**
  - 11 bits
- **a unique number for each PU undergrad**
  - 13 bits

Powers of two, powers of ten

1 bit = 2 possibilities

2 bits = 4 possibilities

3 bits = 8 possibilities

...

n bits = 2^n possibilities

2^{10} = 1,024 is about 1,000 or 1K or 10^3

2^{20} = 1,048,576 is about 1,000,000 or 1M or 10^6

2^{30} = 1,073,741,824 is about 1,000,000,000 or 1G or 10^9

the approximation is becoming less good

but it's still good enough for estimation

- **terminology is often imprecise:**
  - " 1K " might mean 1000 or 1024 (10^3 or 2^{10})
  - " 1M " might mean 1000000 or 1048576 (10^6 or 2^{20})

Bytes

- **"byte" = a group of 8 bits treated as a unit**
  - in modern computers, the fundamental unit of processing and memory addressing
  - can encode any of 2^8 = 256 different values, e.g., numbers 0 .. 255 or a single letter like A or digit like 7 or punctuation like $
    - ASCII character set defines values for letters, digits, punctuation, etc.
- **group 2 bytes together to hold larger entities**
  - two bytes (16 bits) holds 2^{16} = 65,536 values
  - a bigger integer, a character in a larger character set
    - Unicode character set defines values for most characters
- **group 4 bytes together to hold even larger entities**
  - four bytes (32 bits) holds 2^{32} = 4,294,967,296 values
  - an even bigger integer, a number with a fractional part (floating point), a memory address, all Unicode characters
  - current machines use 64-bit integers and addresses (8 bytes)
    - 2^{64} = 18,446,744,073,709,551,616
- **no fractional bytes: the number of bytes is always an integer**

Interpretation of bits and bytes depends on context

- **meaning of a group of bits depends on how they are interpreted**
- **1 byte could be**
  - 1 bit in use, 7 wasted bits (e.g., M/F in a database)
  - 8 bits representing a number between 0 and 255
  - an alphabetic character like W or + or 7
  - part of a character in another alphabet or writing system (2+ bytes)
  - part of a larger number (2 or 4 or 8 bytes, usually)
  - part of a picture or sound
  - part of an instruction for a computer to execute
    - instructions are just bits, stored in the same memory as data
    - different kinds of computers use different bit patterns for their instructions
      - laptop, cellphone, game machine, etc., all potentially different
  - part of the location or address of something in memory
  - ...

- **one program's instructions are another program's data**
  - when you download a new program from the net, it's data
  - when you run it, it's instructions