Lecture 22 Overview

● Signaling constraints
  ◆ Shannon’s Law
  ◆ Nyquist Limit

● Encoding schemes
  ◆ Clock recovery
  ◆ Manchester, NRZ, NRZI, etc.
Ways to measure a channel

- **How fast?**
  - **Bandwidth** measured in bits per second
    - Yes, this is an abuse of terminology—sorry.
  - Often talk about KBps or Mbps – Bytes vs bits

- **How long was the wait?**
  - **Delay** (one-way or round trip) measured in seconds

- **How efficiently?**
  - **Overhead** measured in bits or seconds or cycles or…

- **Any mistakes?**
  - **Error rate** measured in terms of probability of flipped bit
Ok, recall from last class…

- No channel is perfect and the original signal gets modified along the way
  - Attenuation: signal power absorbed by medium
  - Distortion: frequency, phase changes
  - Noise: random background “signals”

- Different mediums distort different signals differently

Note: that here “bandwidth” means frequency over which signals cannot pass through channel
Forms of Digital Modulation

Input Signal

Amplitude Shift Keying (ASK)

Frequency Shift Keying (FSK)

Phase Shift Keying (PSK)

Phase changes
Why Different Schemes?

- Properties of channel and desired application
  - AM vs FM for analog radio

- Efficiency
  - Some modulations can encode many bits for each symbol
    (subject to Shannon limit – more on this next class)

- Aiding with error detection
  - Dependency between symbols… can tell if a symbol wasn’t decoded correctly

- Transmitter/receiver Complexity
Sampling

- To reconstruct a signal, we need to sample it.
Intersymbol Interference

- Bandlimited channels cannot respond faster than some maximum frequency $f$
  - Channel takes some time to settle
- Attempting to signal too fast will mix symbols
  - Previous symbol still “settling in”
  - Mix (add/subtract) adjacent symbols
  - Leads to intersymbol interference (ISI)

- OK, so just how fast can we send symbols?
Speed Limit: Nyquist

- In a channel bandlimited to $f$, we can send at maximum symbol (baud) rate of $2f$ without ISI
Multiple Bits per Symbol

- Nyquist limits the number of symbols per second we can send, but doesn’t talk about the information content in each symbol.

- Couldn’t we send *multiple* bits per symbol:
  - E.g., multiple voltage levels instead of just high/low
  - Four levels gets you two bits, $\log_2 M$ in general (M levels)

- Can combine this observation with Nyquist:
  - *Channel capacity:* $C < 2 B \log_2(M)$

- Why not infinite levels? Infinite bandwidth no?
Noise matters

- Real channels are *noisy* … noise creates measurement challenges

- Example:
  - Encode 4 values using voltage
    - 2 bits per symbol
    - Symbols at 3V, 2V, 1V and 0V
  - What if noise is 0.5V?
    - If you get line level of 2.5V then what symbol is it? 11 or 10?

- Limited to $\sim \log_2 (S/2N)$ bits per symbol
  (S = signal power, N = Noise)
  - Previous example: $S = 3V-0V = 3V$, $N=0.5V$, so we can have $\log_2(3/1) = 1.58$ bits per symbol
Shannon’s Law

- Shannon considered noisy channels and derived

\[ C = B \log (1 + S/N) \]

- Gives us an upper bound on any channel’s performance regardless of signaling scheme

- Old school modems approached this limit
  - \( B = 3000\text{Hz}, \ S/N = 30\text{dB} = 1000 \)
  - \( C = 3000 \times \log(1000) \approx 30\text{kbps} \)
  - 28.8Kbps – anyone remember dialup?
How long to send a message?

- Transmit time $T = \frac{M}{R} + D$
  - 10 Mbps Ethernet LAN (M=1KB)
    - $\frac{M}{R} = \sim 1$ ms, $D = \sim 5$ us
  - 155 Mbps cross country ATM link (M=1KB)
    - $\frac{M}{R} = \sim 50$ us, $D = 40-100$ ms

- Where are the bits in the mean time?
  - In transit inside the network (“in the pipe”)

- $R*D$ is called the **bandwidth-delay product**
  - How many bits can be “stored” be stored in transit
  - Colloquially, we say “fill the pipe”
Next problem: Clock recovery

- How does the receiver know when to sample the signal?
  - Sampling rate: How often to sample?
  - Sampling phase:
    » When to start sampling? (getting in phase)
    » How to adjust sampling times (staying in phase)
Why the sampling rate matters:

- Signal could have multiple interpretations

Which of these is correct?
Nyquist Revisited

- Sampling at the correct rate ($2f$) yields actual signal
  - Always assume lowest-frequency wave that fits samples

- Sampling too slowly yields aliases
The Importance of Phase

- Need to determine when to START sampling, too
Clock Recovery

- Using a training sequence to get receiver lined up
  - Send a few, known initial training bits
  - Adds inefficiency: only $m$ data bits out of $n$ transmitted

- Need to combat clock drift as signal proceeds
  - Use transitions to keep clocks synched up

- Question is, how often do we do this?
  - Quick and dirty every time: asynchronous coding
  - Spend a lot of effort to get it right, but amortize over lots of data: synchronous coding
Asynchronous Coding

- Encode several bits (e.g. 7) together with a leading “start bit” and trailing “stop bit”
- Data can be sent at any time
- Start bit transition kicks of sampling intervals
  - Can only run for a short while before drifting
Example: RS232 serial lines

- Uses two voltage levels (+15V, -15V), to encode single bit binary symbols
- Needs long idle time – limited transmit rate
Synchronous coding

- Encode many bits (thousands) together
  - Amortize cost of learning clock information from start bits (preamble) and stop bits (trailer)
  - Continuously “learn” clock from data stream
    » Watch for 0-1 or 1-0 transitions, and adjust clock
    » Called clock recovery process

- Examples
  - NRZ
  - NRZI
  - Manchester
  - 4B/5B
  - Many others…
For Next Class

- We’ll cover some example synchronous encodings
- Read 2.6