

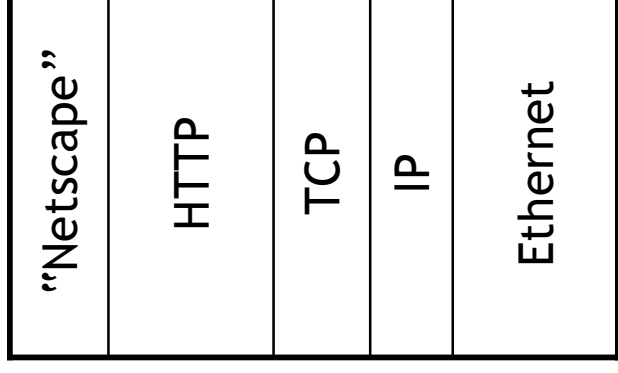
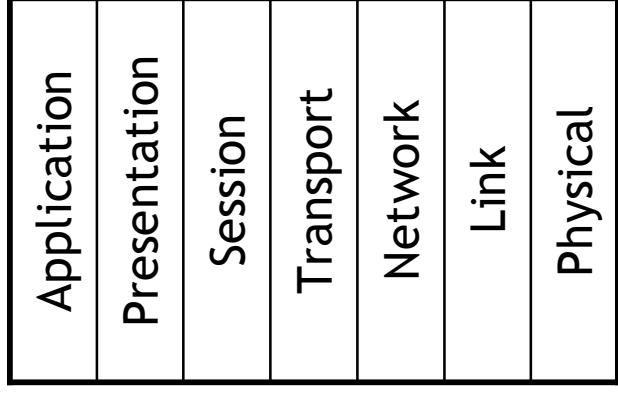
# CSE/EE 461 – Lecture 2



David Wetherall  
djw@cs.washington.edu

# Last Time ...

- We covered protocols, layering and reference models



The OSI Model

Sample Protocol Stack

# This Time



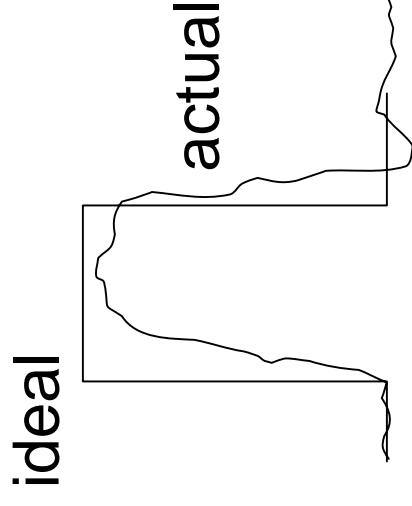
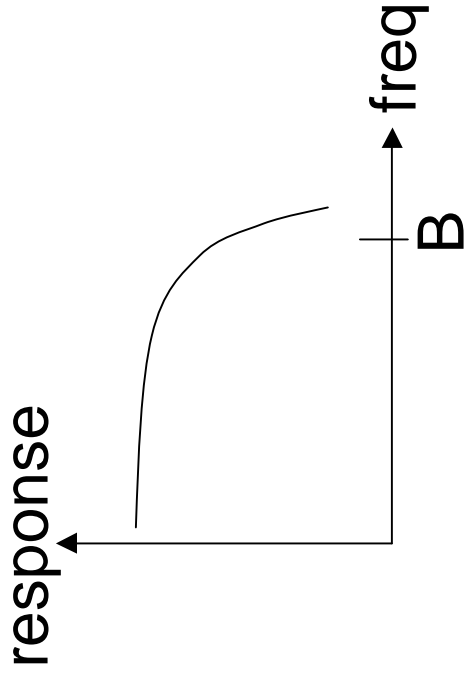
- A look at the physical/link layers:
  1. Media
  2. Limits to transmission
  3. Encoding bits/messages into signals
  4. Error detection and correction
- Key Focus:
  - How do we send a message across a wire?

# 1. Different Kinds of Media

- **Wire**
  - Twisted pair, e.g., CAT5 UTP, 10 → 100Mbps, 100m
  - Coaxial cable, e.g, thin-net, 10 → 100Mbps, 200m
- **Fiber**
  - Multi-mode, 100Mbps, 2km
  - Single mode, 100 → 2400 Mbps, 40km
- **Wireless**
  - Infra-red, e.g., IRDA, ~1Mbps
  - RF, e.g., 802.11 wireless LANs, Bluetooth (2.4GHz)
  - Microwave, satellite, cell phones, ...

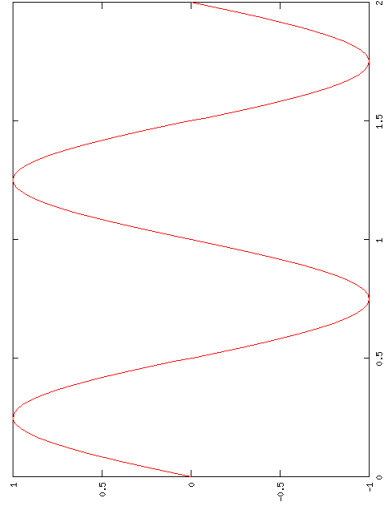
# Wires

- Signal subject to:
  - Attenuation (repeaters)
  - Distortion (frequency and delay)
  - Noise (thermal, crosstalk, impulse)



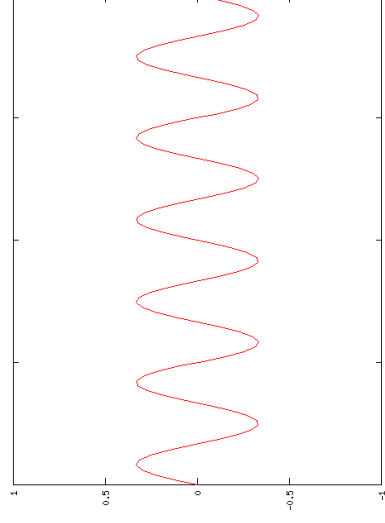
# Example: Square Wave

- Square wave can be decomposed into a sum of sines by a technique called Fourier analysis



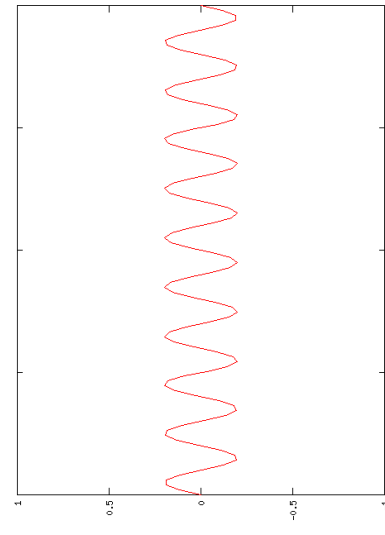
$$\sin(x)$$

+



$$+ \sin(3x)/3$$

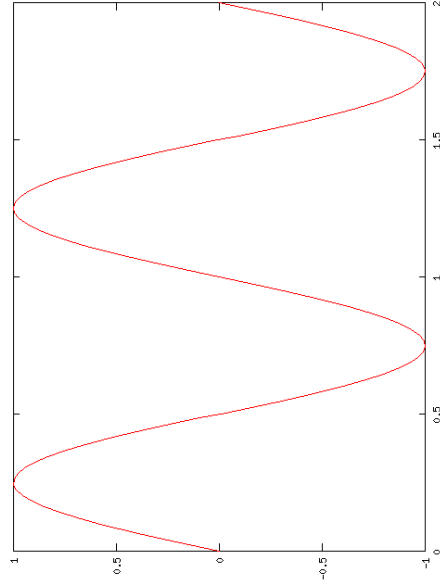
+



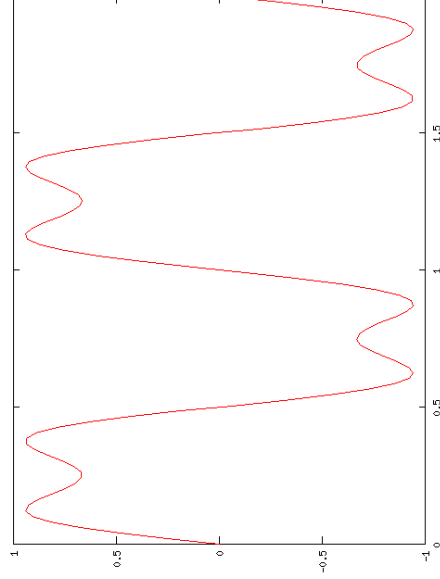
$$+ \sin(5x)/5 \dots$$

# Example: Square Wave

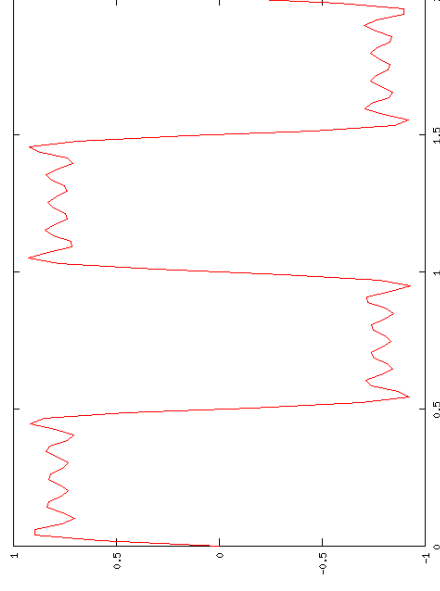
- Adding components back together gives successively better representations allowed by higher bandwidths



fundamental



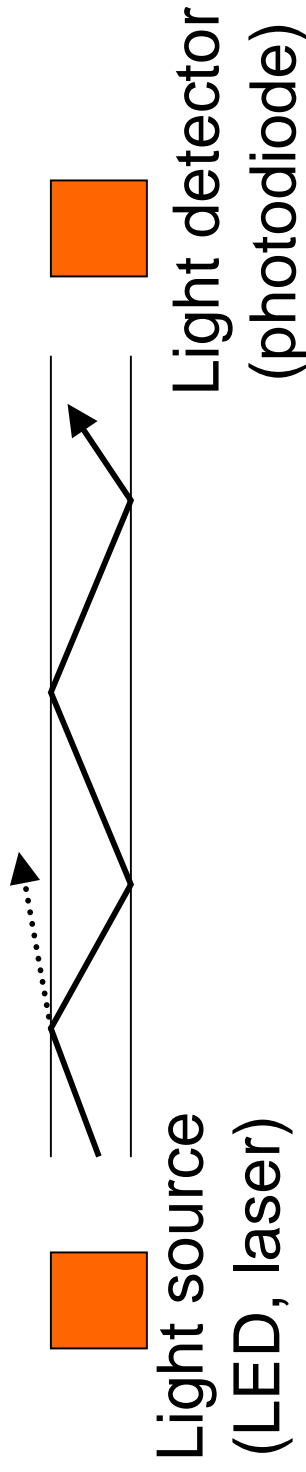
1<sup>st</sup> + 3<sup>rd</sup> harmonic



5 terms ...

# Fiber

- Long, thin, pure strand of glass
  - Enormous bandwidth available (terabits)

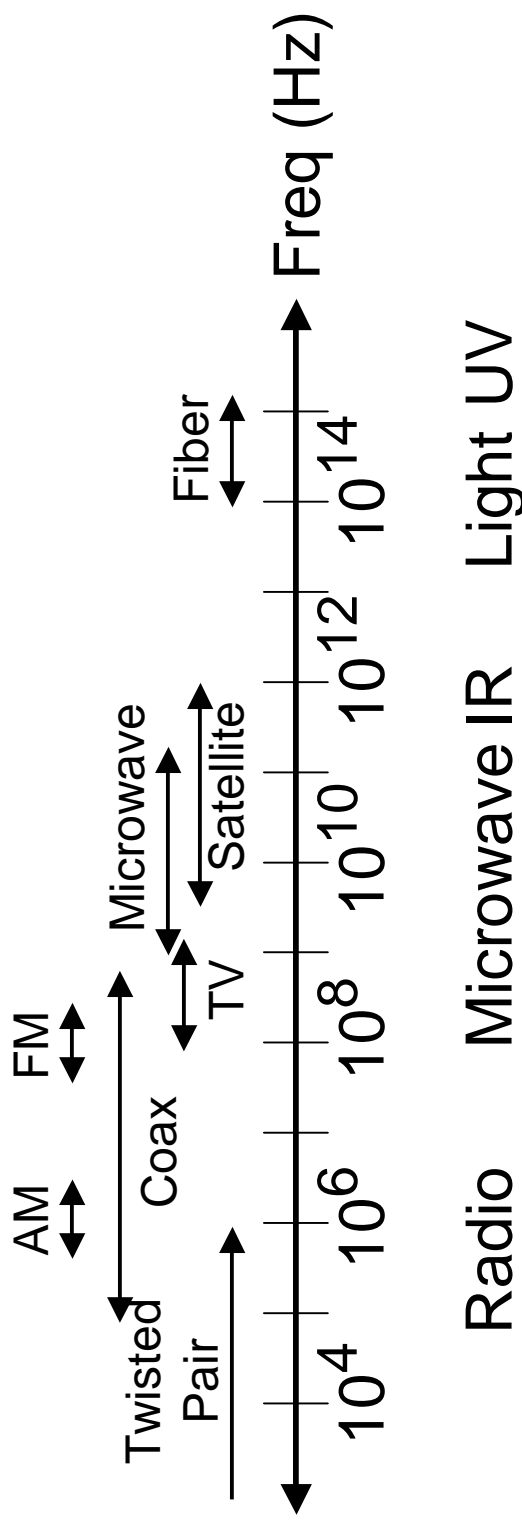


- Multi-mode allows many different paths, dispersion
- Chromatic dispersion if multiple frequencies

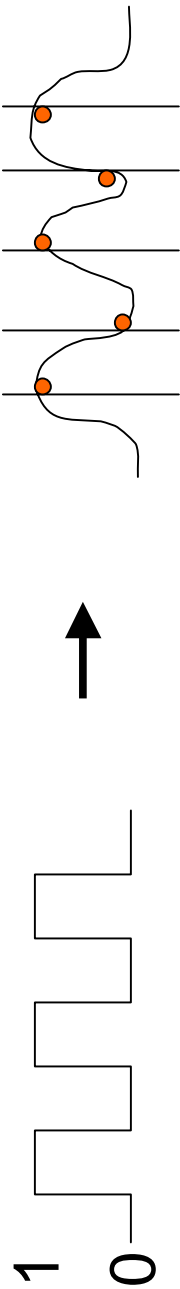


# Wireless

- Different frequencies have different properties
- Signals subject to atmospheric/environmental effects



# Transmission of Digital Data

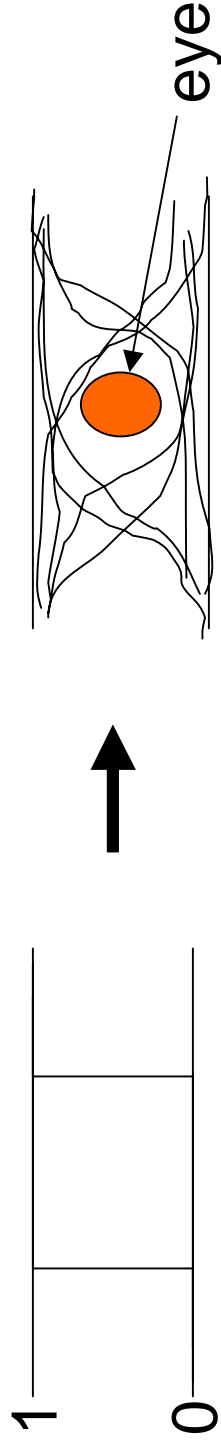
- Digital = discrete; Analog = continuous
  - Generate analog waveform (e.g., voltage) from data at transmitter and sample to recover at receiver
- 
- We send/recover symbols that are mapped to bits (coming later)
    - Signal transition rate = baud rate, versus bit rate
  - This is baseband transmission. Data modulated onto carriers for broadband transmission, e.g., AM/FM radio
  - I've omitted \*tons\*. Take a signals course!

## 2. Limits to Transmission

- How fast can we send bits down a wire?
  - Limited by bandwidth → Nyquist
  - Limited by noise → Shannon
- These are two fundamental results

# Nyquist Limit ( $\sim 1924$ )

- For a noiseless channel with bandwidth  $B$
- Symbols will be distorted, and sending too fast leads to Inter-symbol Interference (ISI)



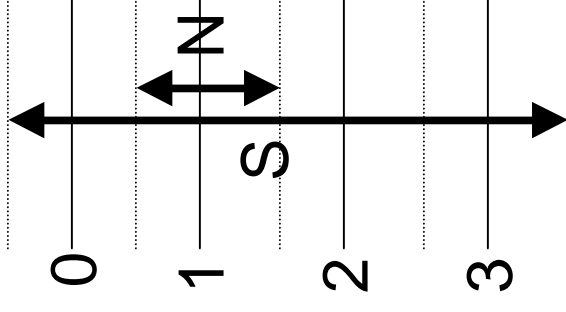
- The maximum rate at which it is possible to send is:

$$R = 2B \text{ symbols/sec}$$

e.g., 3KHz  $\rightarrow$  6Ksym/sec

# Taking Noise into Account

- Noise limits how many signal levels we can safely distinguish between
  - $S$  = max signal amp.,  $N$  = max noise amp.
- The number of bits per symbol depends on the number of signal levels
  - E.g, 4 levels implies 2 bits / symbol

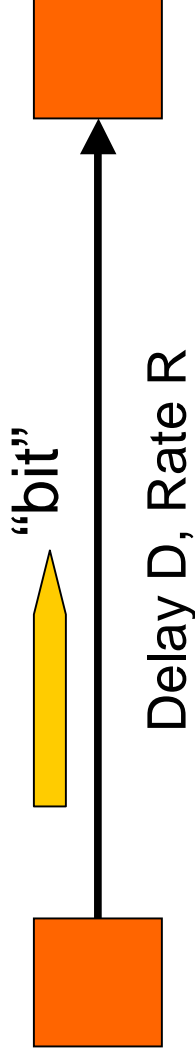


# Shannon Limit (1948)

- Noise expressed as Signal to Noise Ratio (SNR)
  - $SNR = 10\log_{10}(\text{signal} / \text{noise})$  decibels (dB)
    - e.g, 30 dB means signal 1000 times as powerful as noise
- For a noisy channel with given SNR and bandwidth B, the maximum rate at which it is possible to send is:
  - $C = B \log_2(1 + SNR)$  bits/sec
    - e.g 3KHz and 30dB SNR  $\rightarrow$  30Kbps
- This is the information carrying capacity of the channel. Bandwidth is often a loosely used term.

# A Note on Latency

- How long does it take to send a message?



- Two terms:
  - Propagation delay is a function(length of wire)
    - $D = \text{distance} / \text{speed of light in media}$
  - Transmission delay is a function(encoding, message length)
    - $\text{TX} = \text{message (bits)} / \text{rate (bps)}$
    - In effect, slow links stretch bits out in time/space
  - Later we will see queuing delay ...

# One-way Latency Examples

- Either a slow link or long wire makes for large latency
- Dialup with a modem:
  - $D = 10\text{ms}$  (say),  $R = 56\text{Kbps}$ ,  $M = 1000$  bytes
  - Latency =  $10\text{ms} + (1024 \times 8) / (56 \times 1024) \text{ sec} = 153\text{ms!}$
- Cross-country with T3 line:
  - $D = 50\text{ms}$ ,  $R = 45\text{Mbps}$ ,  $M = 1000$  bytes
  - Latency =  $50\text{ms} + (1024 \times 8) / (45 \times 1000000) \text{ sec} = 50\text{ms!}$



# 3. Encoding Bits

- How to represent bits as signals?
- We will look at two problems:
  - Clock recovery
    - When to sample the signal to recover bits
  - Framing
    - How to recognize the when a message starts

# NRZ and NRZI

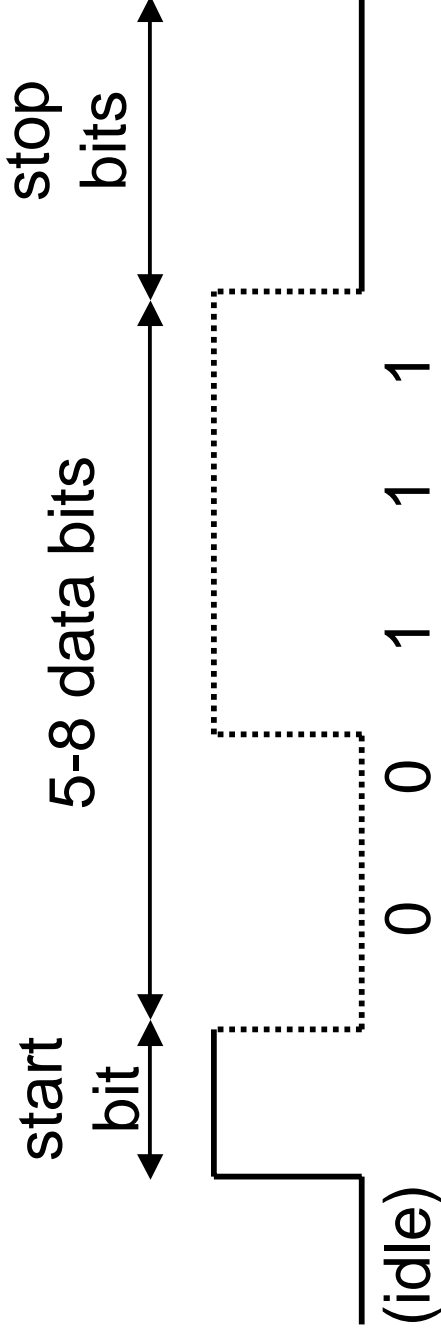
- Simplest scheme, NRZ (Non-return to zero)
  - Use high/low voltages, e.g., high = 1, low = 0
- Variation, NRZI (NRZ, invert on 1)
  - Use transition for 1s, no transition for 0s
- (example following)

# Clock Recovery

- Problem: How do we distinguish consecutive 0s or 1s?
- If we sample at the wrong time we get garbage ...
- If sender and receiver have exact clocks no problem
  - But in practice they drift slowly
- This is the problem of clock recovery
- Possible solutions:
  - Send separate clock signal → expensive
  - Keep messages short → limits data rate
  - Embed clock signal in data signal → other codes

# "Asynchronous" Transmission

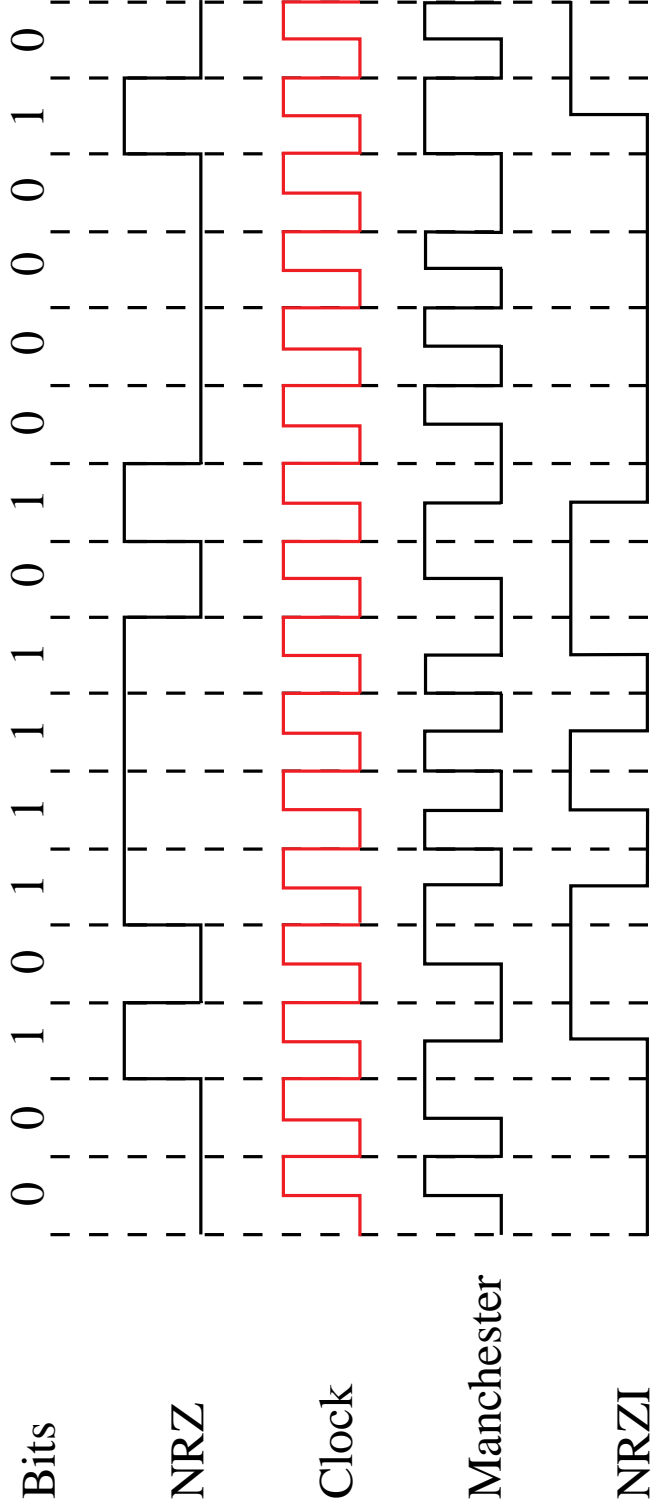
- Avoid timing problem by sending short, delimited data
  - E.g., UARTs (typically used to connect your keyboard)



# Manchester Coding

- Make transition in the middle of every bit period
  - Low-to-high is 0; high-to-low is 1
  - Signal rate is twice the bit rate
  - Used on 10 Mbps Ethernet
- (example following)
- Advantage: self-clocking
- Disadvantage: 50% efficiency

# Coding Examples



# 4B/5B Codes

- We want transitions \*and\* efficiency ...
- Solution: map data bits (which may lack transitions) into code bits (which are guaranteed to have them)
- 4B/5B code:
  - 0000 → 11110, 0001 → 01001, ... 1111 → 11101
  - Never more than three consecutive 0s back-to-back
  - 80% efficiency
- This code is used by LANs such as FDDI

# Framing

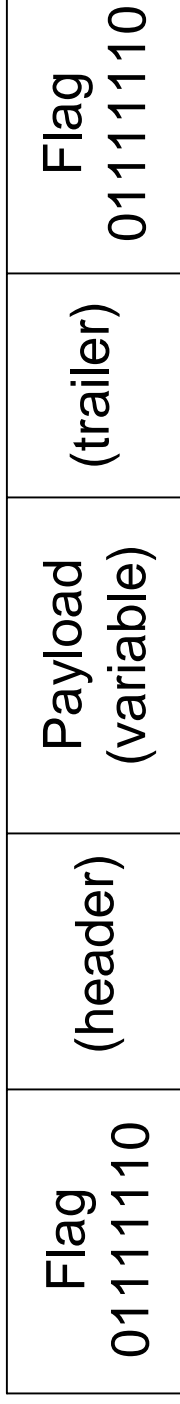


- Need to send message, not just bits
  - Even if we know where the bits are we still need to synchronize on the start of the message
  - Complete Link layer messages are called frames
- Common approach: Sentinels
  - Look for special control code that marks start of frame



# Point-to-Point Protocol (PPP)

- IETF standard, used for dialup and leased lines



- Flag indicates start/end of frame
- Occurrences of flag inside payload must be “stuffed”
  - Replace “flag” with “flag flag”
  - Length of payload is data-dependent!

# Other Approaches

- Use explicit byte count after preamble
  - More susceptible to errors?
- Use “invalid” codeword
  - E.g., pick non-data 4B/5B symbol, used for FDDI
- SONET: “clock”-based framing
  - Periodic sync bits plus very accurate clock
  - Used extensively in the telecommunications industry

# 4. Error Detection / Correction

- Noise can flip some of the bits we receive
  - At a minimum we must be able to detect when this occurs
- Basic approach: add redundant data
  - Error detection codes allow errors to be recognized
  - Error correction codes allow some errors to be repaired too

# Motivating Example

- Let's just send two copies. Differences imply errors.
- Question: Can we do any better?
  - With less overhead
  - Catch more kinds of errors
- Answer: Yes - stronger protection with fewer bits
  - But we can't catch all inadvertent errors, nor malicious ones
- We will look at basic block codes
  - $K$  bits in,  $N$  bits out is a  $(N,K)$  code
  - Simple, memoryless mapping

# Detection versus Correction

- Two strategies to correct errors:
  - Error correcting codes and retransmissions (ARQ)
- Question: Which should we choose?
- Answer: Depends on errors and cost of recovery!
- Example: Message with 1000 bits, Prob(bit error) 0.001
  - If random errors, most messages likely to have an error
  - If bursts of 1000 errors typical, only 1 or 2 per 1000 messages
- Satellites, real-time media tend to use error correction
  - Called Forward Error Correction (FEC) in some contexts
- Retransmissions typically at the frame/packet level

# The Hamming Distance

- To detect/correct bit errors, errors must not turn one valid codeword into another valid codeword
- Hamming distance is the number of bit differences
  - E.g, code 000 for 0, 111 for 1, Hamming distance is 3
  - This is the number of errors needed to turn one into the other
  - Hamming distance of the entire code is minimum of pairs
- For code with distance  $d+1$ :
  - $d$  errors can be detected, e.g, 001, 010, 110, 101, 011
- For code with distance  $2d+1$ :
  - $D$  errors can be corrected, e.g., 001  $\rightarrow$  000 iff one error

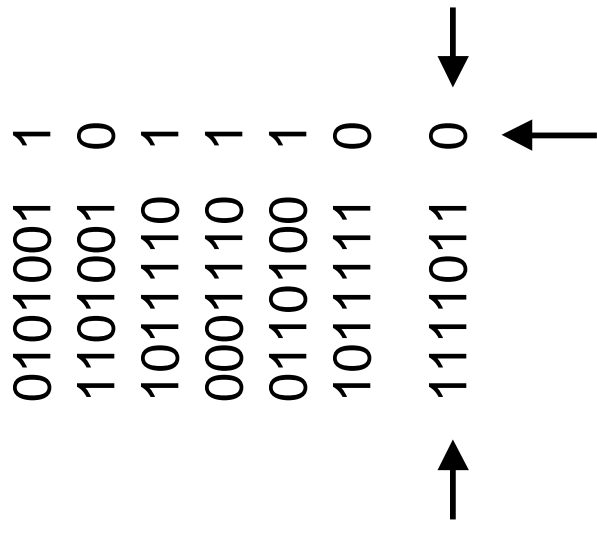
# Parity

- Start with  $n$  bits and add another so that the total number of 1s is even (even parity)
  - e.g. 0110010  $\rightarrow$  01100101
  - Easy to compute as XOR of all input bits
- Will detect an odd number of bit errors
  - But not an even number
- Does not correct any errors

# 2D Parity

- Add parity row/column to array of bits

- Detects all 1, 2, 3 bit errors
- Corrects all 1 bit errors





# Checksums



- Used in Internet protocols (IP, ICMP, TCP, UDP)
- Basic Idea: Add up the data and send it along with sum
- Algorithm:
  - checksum is the 1s complement of the 1s complement sum of the data interpreted 16 bits at a time (for 16-bit TCP/UDP checksum)
- 1s complement: flip all bits to make number negative
  - Consequence: adding requires carryout to be added back

# Checksum Example

- Message is e3 4f 23 96 44 27 99 f3
- 2s complement sum is 1e4ff
- So 1s complement sum is e500 (add back carry)
- So checksum is 1aff (flip all bits)
  
- Advantages: fast to compute; incremental
- Disadvantage: error detection isn't strong

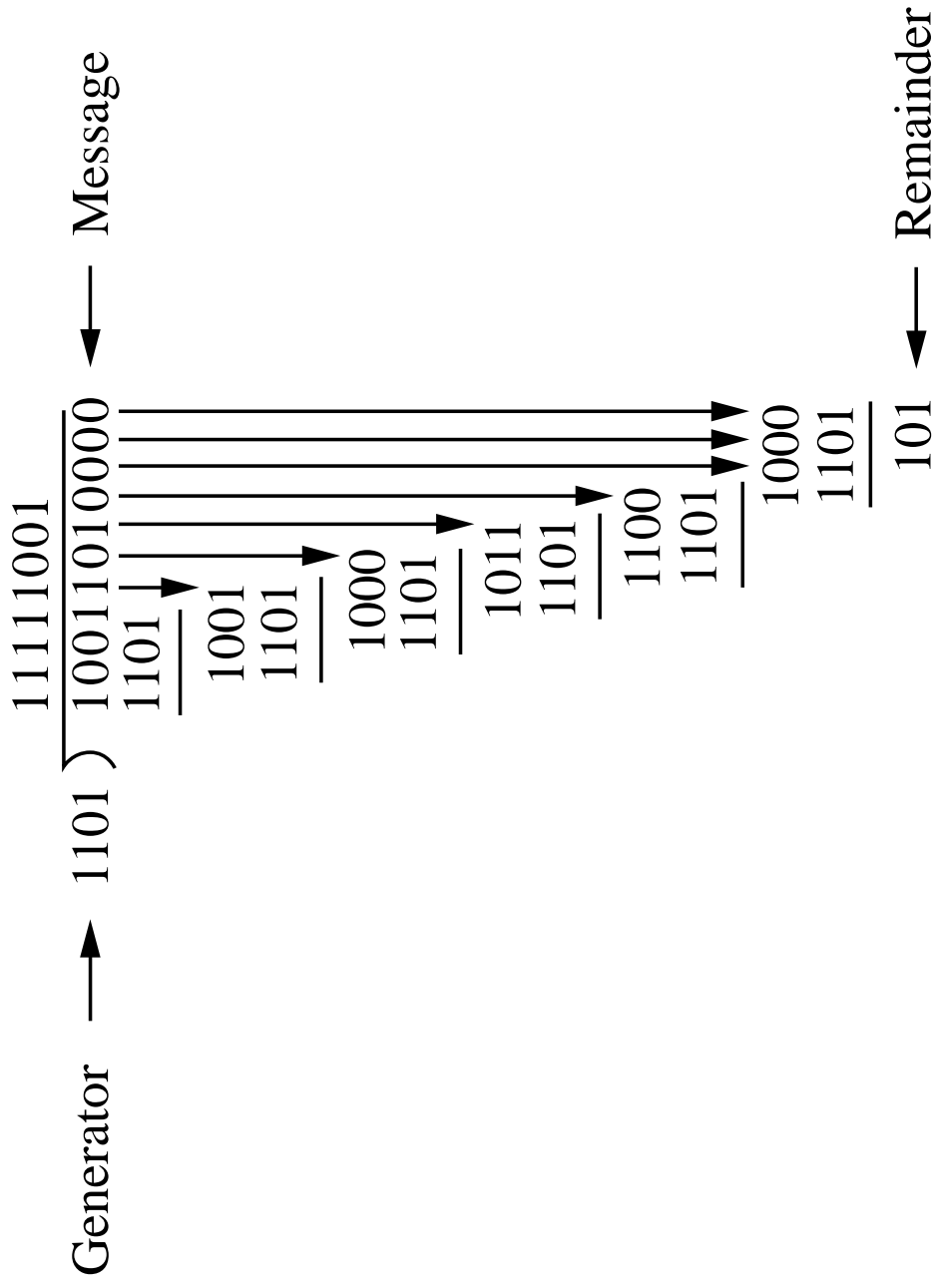
# CRCs (Cyclic Redundancy Check)

- Stronger protection than checksums
  - Used widely in practice, e.g., Ethernet CRC-32
  - Easily implemented in hardware (XORs and shifts)
- Algorithm: Given  $n$  bits of data, generate a  $k$  bit check sequence that gives a combined  $n + k$  bits that are divisible by a pre-defined number
- Based on mathematics of finite fields
  - “numbers” correspond to polynomials, use modulo arithmetic
  - e.g, interpret 10011010 as  $x^7 + x^4 + x^3 + x^1$

# CRC Example

- How do we generate the check sequence?
  - Have our message, e.g., 10011010 (m=8)
  - Have the divisor polynomial, e.g.,  $C(x)=1110$  ( $x^3 + x^2 + x^1 + x^0$ ; k=3)
  - Want to make m + k bits divisible by this divisor ...
  - First, add k zeros to end of message
  - Then, divide by C(x) to find the remainder ...

# Example – Polynomial Division



# Example – Remainder to CRC

- So we see the remainder is 101
- Thus the zero extended message - 101 must be evenly divisible by  $C(x)$ !
- So perform the subtraction to discover the check bits
  - Subtraction/addition is XOR in modulo 2 arithmetic
  - E.g., we get  $10011010000 - 101 = 1011010101$
  - The check bits are 101

# How is $C(x)$ Chosen?

- Mathematical properties:
  - All 1-bit errors if non-zero  $x^k$  and  $x^0$  terms
  - All 2-bit errors if  $C(x)$  has a factor with at least three terms
  - Any odd number of errors if  $C(x)$  has  $(x + 1)$  as a factor
  - Any burst error  $< k$  bits
- There are standardized polynomials of different degree that are known to catch many errors

# Standard CRC Polynomials

- CRC-8    100000111
- CRC-10    11000110011
- CRC-12    11000000111
- CRC-16    1000100000100000
- CRC-32    100000100110000010001110110110111



# Reed-Solomon / BCH Codes

- Reed-Solomon codes developed to protect data on magnetic disks
- Used for CDs and cable modems too
- Property:  $2t$  redundant bits can correct  $\leq t$  errors
- Mathematics somewhat more involved ...

# Key Concepts



- Different media have different properties
  - Affects higher layer protocols
- There are fundamental limits to transmission
  - Nyquist and Shannon limits
- To send messages in practice we must solve the problems of clock recovery and framing
- And we must be able to at least detect errors