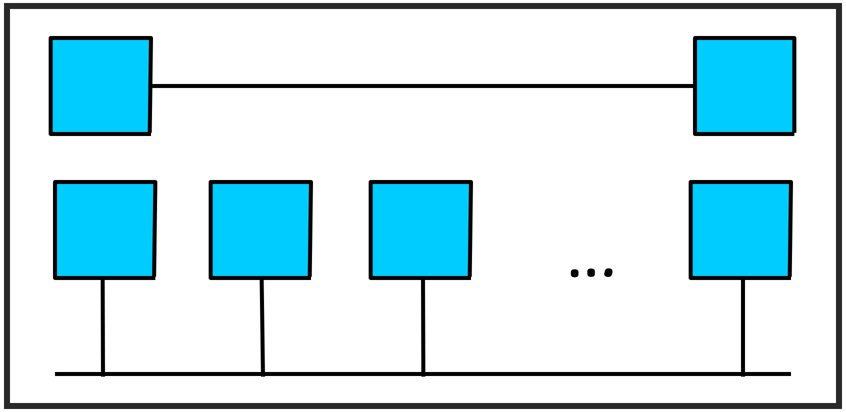
Direct Link Networks

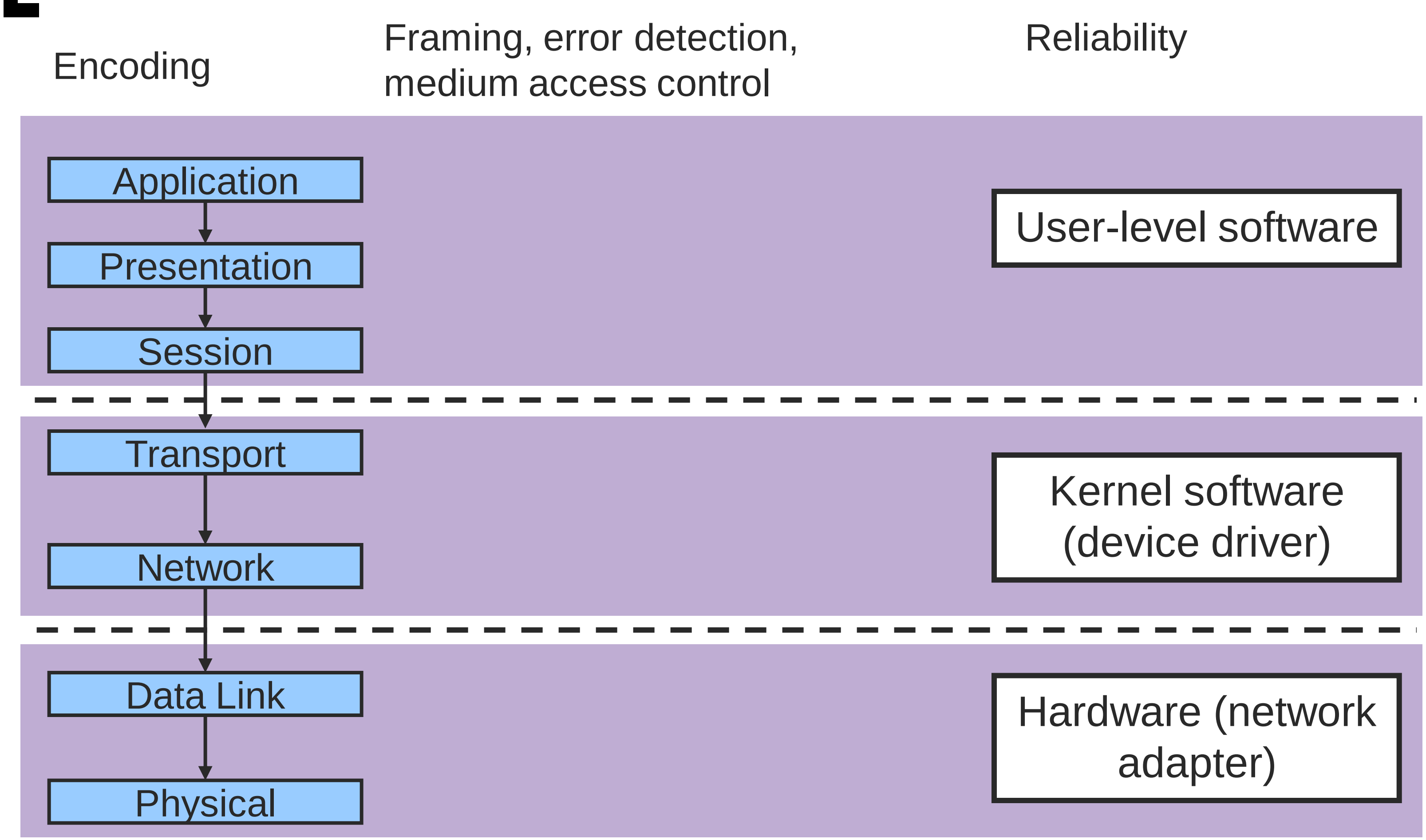
Reading: Peterson and Davie, Chapter 2

Direct Link Networks

- All hosts are directly connected by a physical medium
- Key points
  - Encoding and Modulation
  - Framing
  - Error Detection
  - Reliable Transmission
  - Medium Access Control



Internet Protocols



Direct Link Networks - Outline

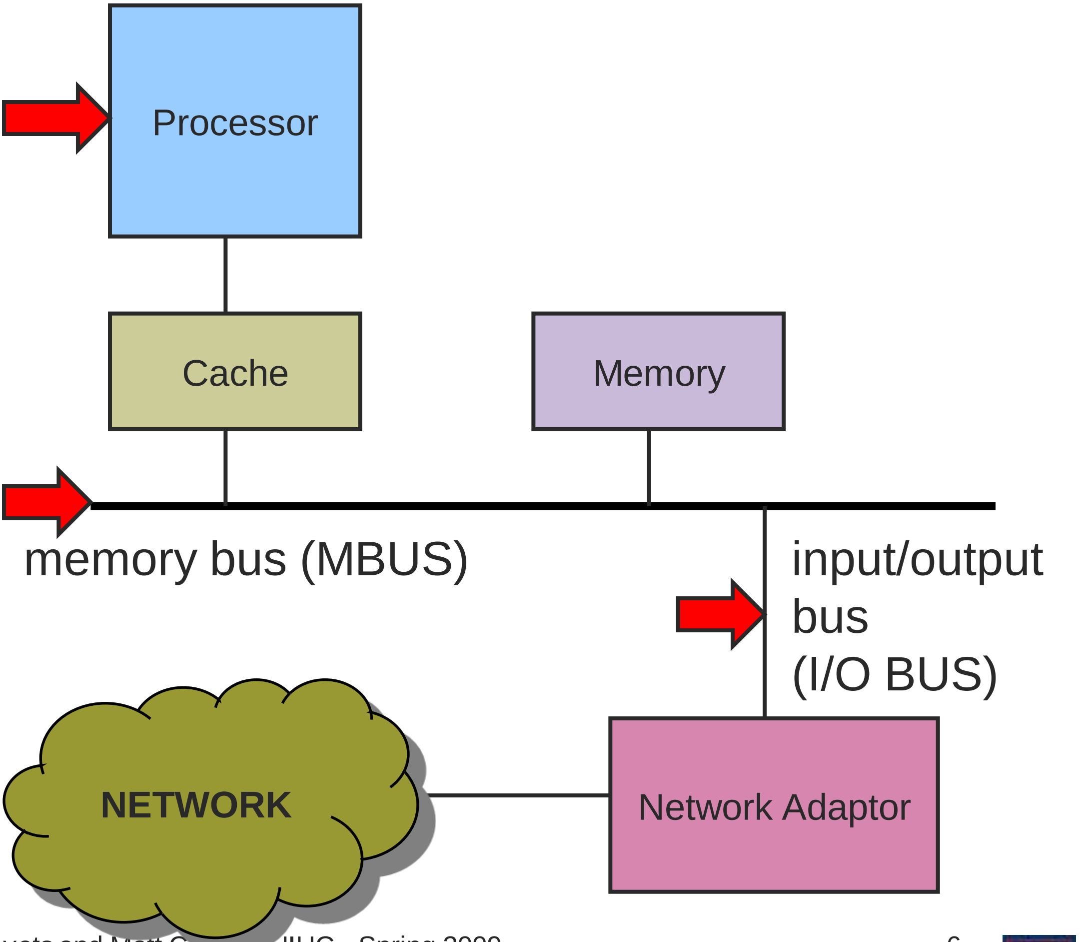
- Hardware building blocks
- Encoding
- Framing
- Error detection
- Reliable transmission
- Multiple access media (MAC examples)
- Network adapters

Hardware Building Blocks

- Nodes
  - Hosts: general purpose computers
  - Switches: typically special purpose hardware
  - Routers: varied

Nodes: Workstation Architecture

- Finite memory
  - Scarce resource
- Generally limited by bus speeds, NOT processor speeds



Hardware Building Blocks

- Links
  - Physical medium carrying
  - Media
    - Copper wire with electronic signaling
    - Glass fiber with optical signaling
    - Wireless with electromagnetic (radio, infrared, microwave) signaling

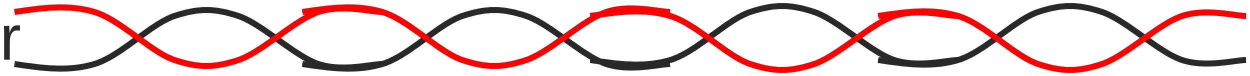
Links - Copper

more twists, less crosstalk, better signal over longer distances

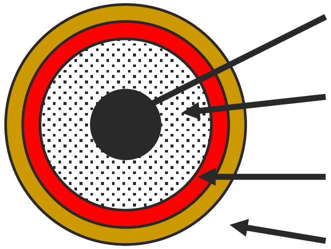
- Copper-based Media

Category 3 Twisted Pair	up to 100 Mbps	
Category 5 Twisted Pair	10-100Mbps	100m
ThinNet Coaxial Cable	10-100Mbps	200m
ThickNet Coaxial Cable	10-100Mbps	500m

twisted pair



coaxial cable (coax)

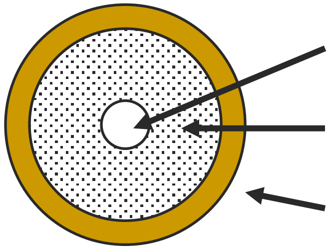


More expensive than twisted pair
High bandwidth and excellent noise immunity

Links - Optical

- Optical Media

Multimode Fiber	100Mbps	2km
Single Mode Fiber	100-2400Mbps	40km



optical fiber

glass core (the fiber)

glass cladding

plastic jacket

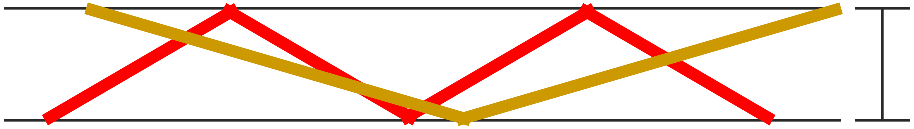
Links - Optical

- Single mode fiber
  - Expensive to drive (Lasers)
  - Lower attenuation (longer distances) ≤ 0.5 dB/km
  - Lower dispersion (higher data rates)
- Multimode fiber
  - Cheap to drive (LED’s)
  - Higher attenuation
  - Easier to terminate

core of single mode fiber

~1 wavelength thick =
~1 micron

core of multimode fiber (same frequency; colors for clarity)



O(100 microns) thick

Links - Optical

- Advantages of optical communication
  - Higher bandwidths
  - Superior attenuation properties
  - Immune from electromagnetic interference
  - No crosstalk between fibers
  - Thin, lightweight, and cheap (the fiber, not the optical-electrical interfaces)

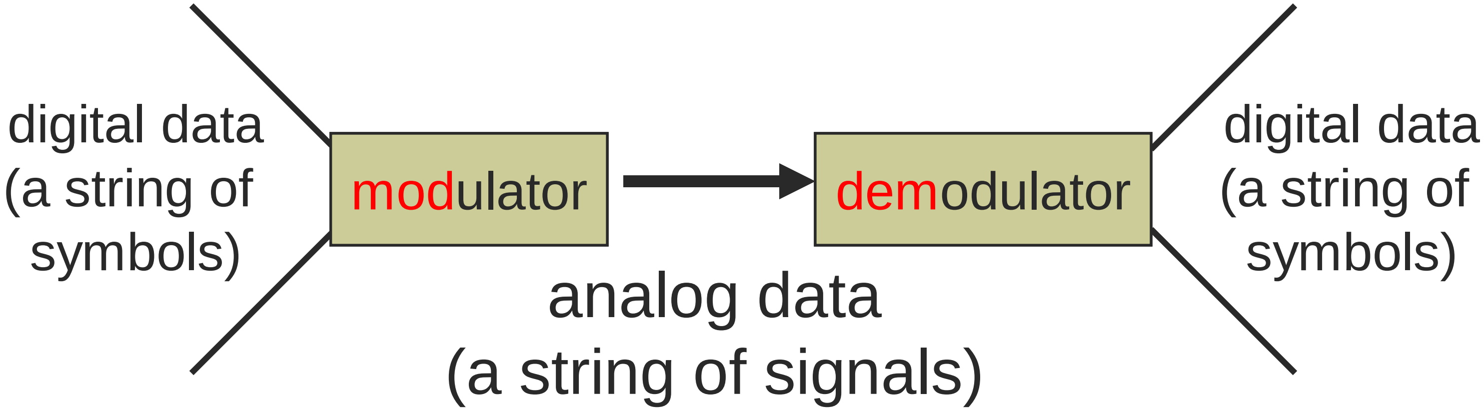
Leased Lines

POTS	64Kbps
ISDN	128Kbps
ADSL	1.5-8Mbps/16-640Kbps
Cable Modem	0.5-2Mbps
DS1/T1	1.544Mbps
DS3/T3	44.736Mbps
STS-1	51.840Mbps
STS-3 (ATM rate)	155.250Mbps (ATM)
STS-12 (ATM rate)	622.080Mbps (ATM)
OC-48	2.5 Gbps

Wireless

Cellular		
AMPS	13Kbps	3km
PCS, GSM	300Kbps	3km
Wireless Local Area Networks (WLAN)		
Infrared	4Mbps	10m
900Mhz	2Mbps	150m
2.4GHz	2Mbps	150m
2.4Ghz	11Mbps	80m
2.4Ghz	54Mbps	75m
5Ghz	54Mbps	30m
Bluetooth	700Kbps	10m
Satellites		
Geosynchronous satellite	600-1000 Mbps	continent
Low Earth orbit (LEO)	~400 Mbps	world

Encoding



- Problems with signal transmission
    - Attenuation: Signal power absorbed by medium
    - Dispersion: A discrete signal spreads in space
    - Noise: Random background “signals”

Encoding

- Goal
  - Understand how to connect nodes in such a way that bits can be transmitted from one node to another
- Idea
  - The physical medium is used to propagate signals
    - Modulate electromagnetic waves
    - Vary voltage, frequency, wavelength
  - Data is encoded in the signal

Analog vs. Digital Transmission

- Analog and digital correspond roughly to continuous and discrete
- Data: entities that convey meaning
  - Analog: continuously varying patterns of intensity (e.g., voice and video)
  - Digital: discrete values (e.g., integers, ASCII text)
- Signals: electric or electromagnetic encoding of data
  - Analog signal: continuously varying electromagnetic wave
    - May be propagated over a variety of medium
  - Digital signal: sequence of voltage pulses
    - May be transmitted over a wire medium

Analog vs. Digital Transmission

- Advantages of digital transmission over analog
  - Cheaper
  - Suffers more attenuation
    - But reasonably low-error rates over arbitrary distances
    - Calculate/measure effects of transmission problems
    - Periodically interpret and regenerate signal
  - Simpler for multiplexing distinct data types (audio, video, e-mail, etc.)
  - Easier to encrypt
- Two examples based on modulator-demodulators (modems)
  - Electronic Industries Association (EIA) standard: RS-232
  - International Telecommunications Union (ITU)
    V.32 9600 bps modem standard

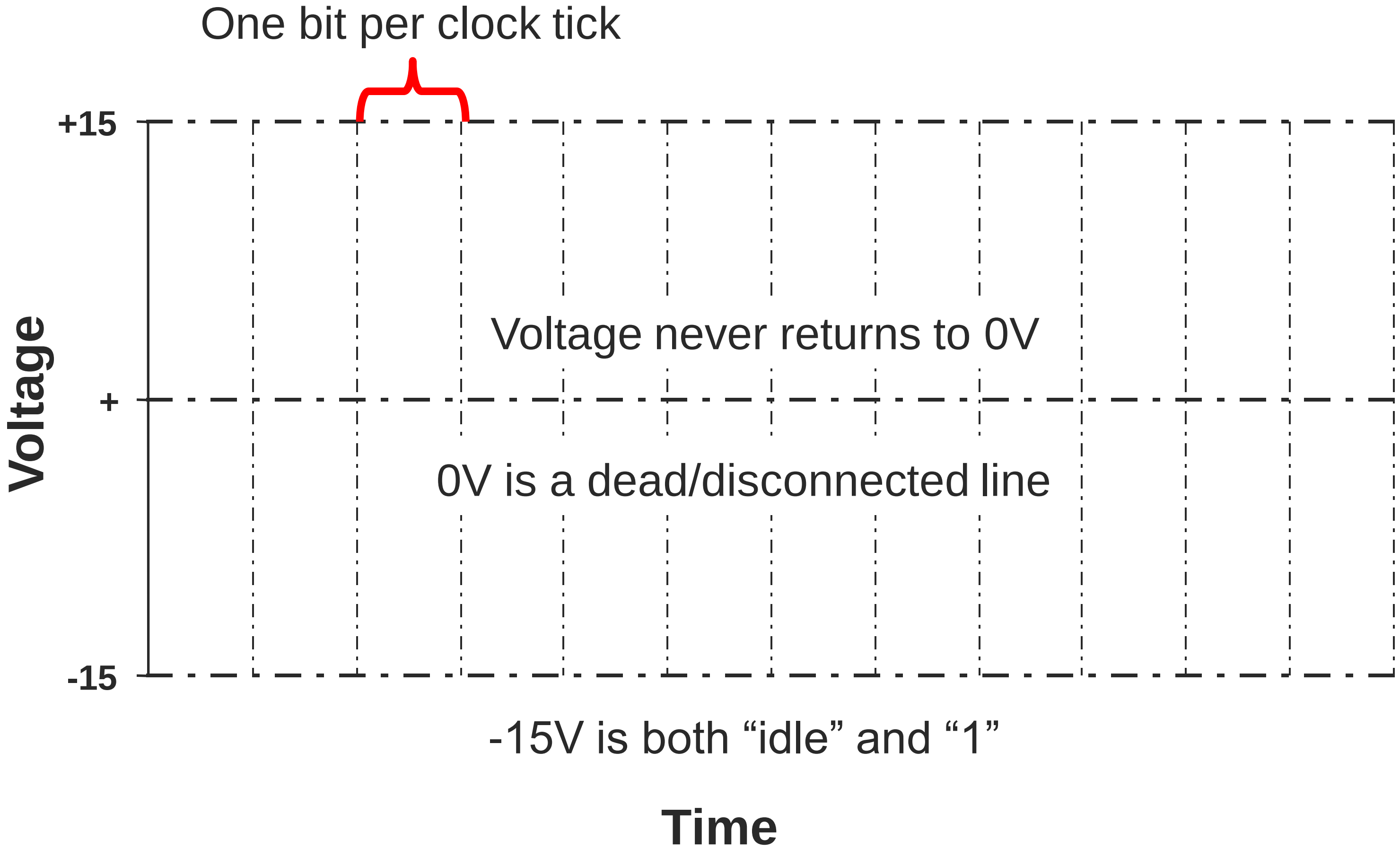
Bauds and Bits

- Baud rate
  - Number of symbols transmitted per second
- Bit rate
  - Actual number of bits transmitted per second
- Relationship
  - Depends on the number of bits encoded in each symbol

RS-232

- Communication between computer and modem
- Uses two voltage levels (+15V, -15V),
  a binary voltage encoding
- Data rate limited to 19.2 kbps (RS-232-C); raised in later standards
- Characteristics
  - Serial
    - One signaling wire, one bit at a time
  - Asynchronous
    - Line can be idle, clock generated from data
  - Character-based
    - Send data in 7- or 8-bit characters

RS-232 Timing Diagram



Voltage never returns to 0V

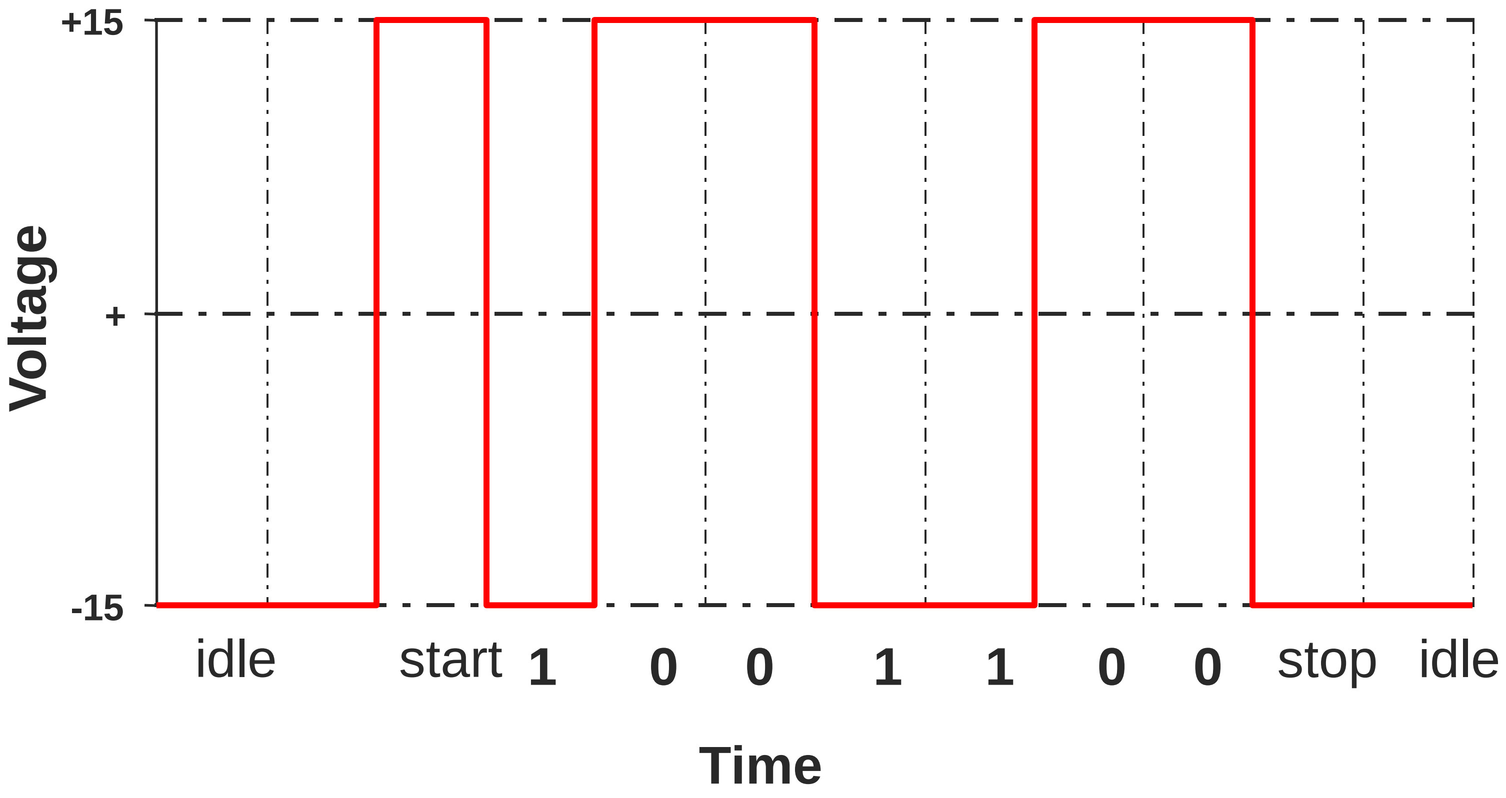
0V is a dead/disconnected line

-15V is both “idle” and “1”

RS-232

- Initiate send by
  - Push to 15V for one clock (start bit)
- Minimum delay between character transmissions
  - Idle for one clock at -15V (stop bit)
- One character
  - 2+ voltage transitions
- Total Bits
  - 9 bits for 7 bits of data (78% efficient)
- Start and stop bits also provide framing

RS-232 Timing Diagram

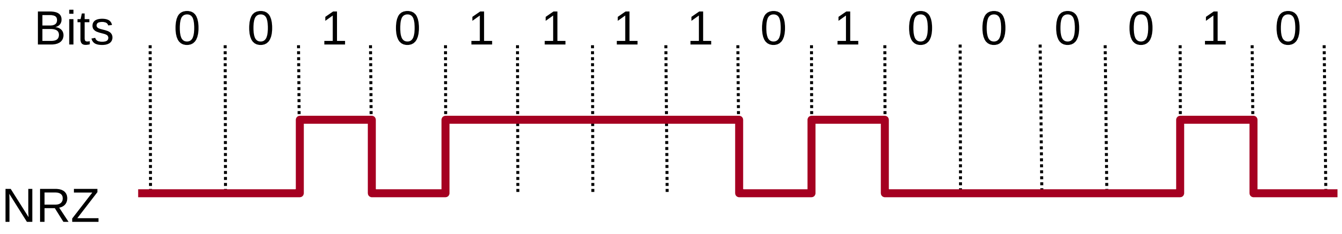


Voltage Encoding

- Binary voltage encoding
  - Done with RS-232 example
  - Generalize before continuing with V.32 (not a binary voltage encoding)
- Common binary voltage encodings
  - Non-return to zero (NRZ)
  - NRZ inverted (NRZI)
  - Manchester (used by IEEE 802.3—10 Mbps Ethernet)
  - 4B/5B

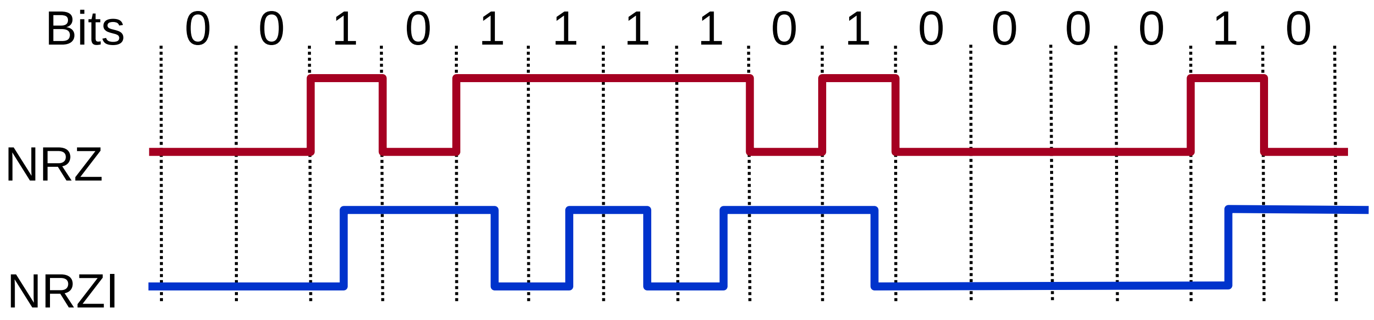
Non-Return to Zero (NRZ)

- Signal to Data
  - High ⇨ 1
  - Low ⇨ 0
- Comments
  - Transitions maintain clock synchronization
  - Long strings of 0s confused with no signal
  - Long strings of 1s causes baseline wander
  - Both inhibit clock recovery



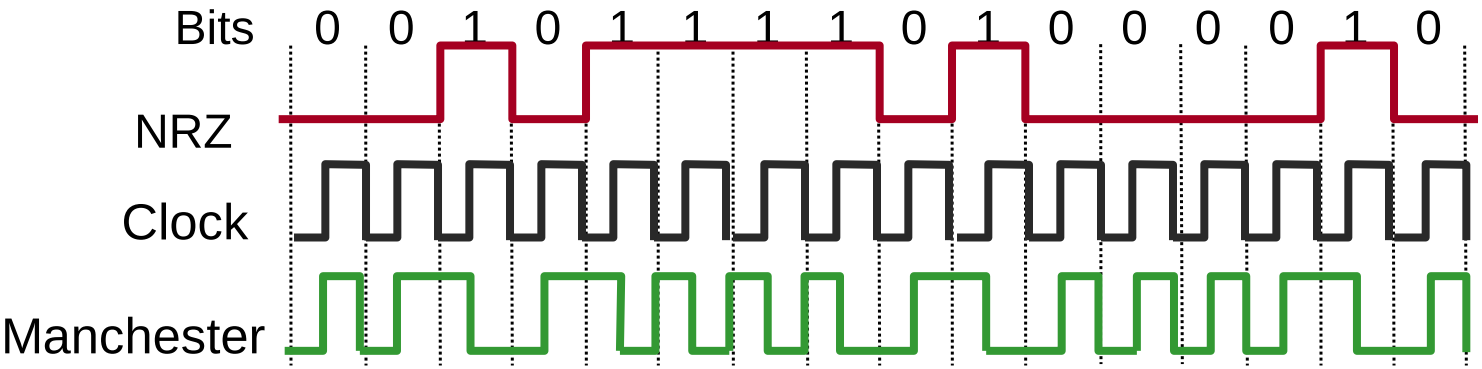
Non-Return to Zero Inverted (NRZI)

- Signal to Data
  - Transition ⇨ 1
  - Maintain ⇨ 0
- Comments
  - Solves series of 1s, but not 0s



Manchester Encoding

- Signal to Data
  - XOR NRZ data with clock
  - High to low transition ⇨ 1
  - Low to high transition ⇨ 0
- Comments
  - (used by IEEE 802.3—10 Mbps Ethernet)
  - Solves clock recovery problem
  - Only 50% efficient (½ bit per transition)



4B/5B

- Signal to Data
  - Encode every 4 consecutive bits as a 5 bit symbol
- Symbols
  - At most 1 leading 0
  - At most 2 trailing 0s
  - Never more than 3 consecutive 0s
  - Transmit with NRZI
- Comments
  - 16 of 32 possible codes used for data
  - At least two transitions for each code
  - 80% efficient

4B/5B – Data Symbols

At most 1 leading 0

- 0000 ⇒ 11110
- 0001 ⇒ 01001
- 0010 ⇒ 10100
- 0011 ⇒ 10101
- 0100 ⇒ 01010
- 0101 ⇒ 01011
- 0110 ⇒ 01110
- 0111 ⇒ 01111

At most 2 trailing 0s

- 1000 ⇒ 10010
- 1001 ⇒ 10011
- 1010 ⇒ 10110
- 1011 ⇒ 10111
- 1100 ⇒ 11010
- 1101 ⇒ 11011
- 1110 ⇒ 11100
- 1111 ⇒ 11101

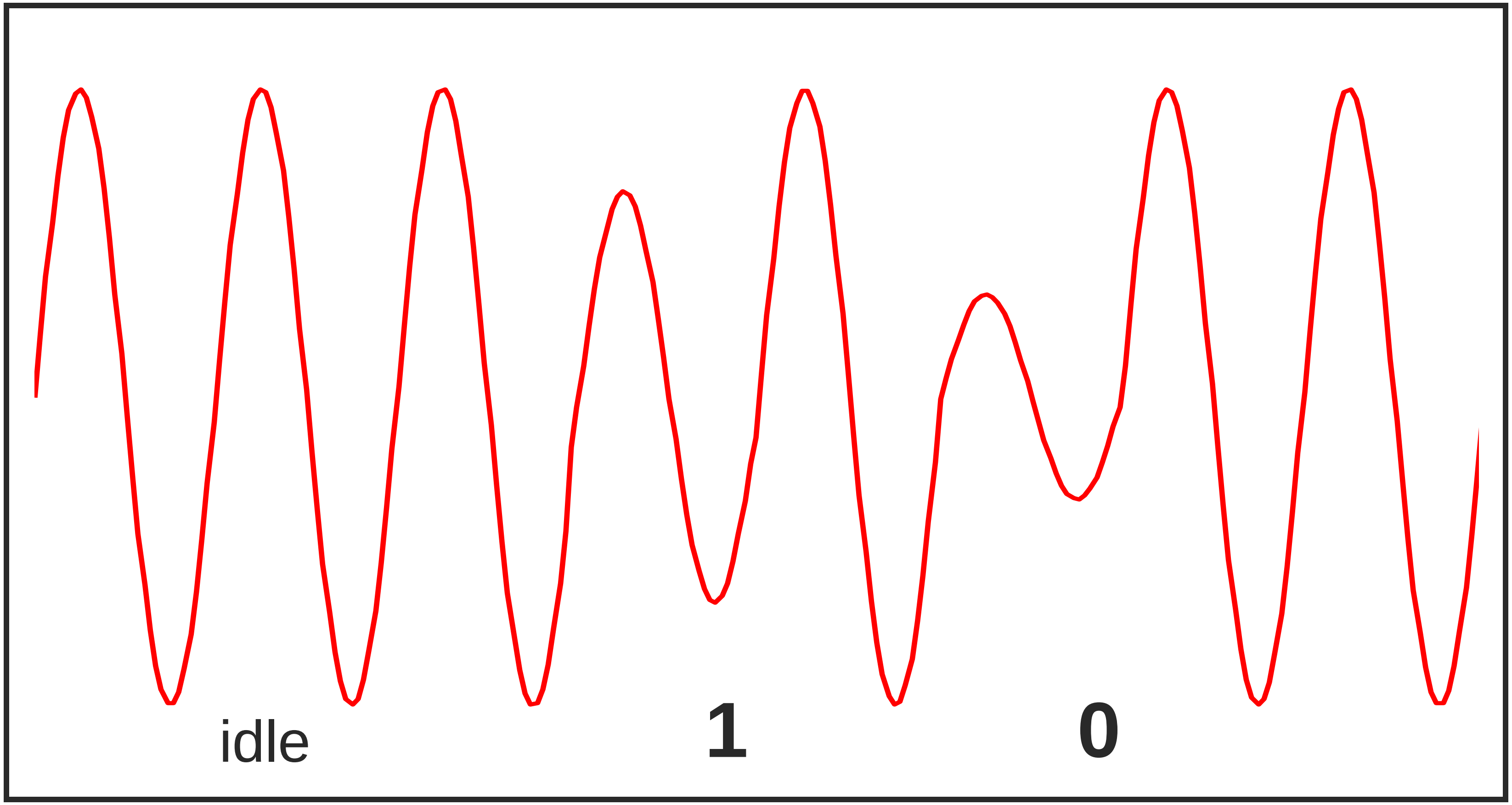
4B/5B – Control Symbols

- 11111 ⇒ idle
- 11000 ⇒ start of stream 1
- 10001 ⇒ start of stream 2
- 01101 ⇒ end of stream 1
- 00111 ⇒ end of stream 2
- 00100 ⇒ transmit error
- Other ⇒ invalid

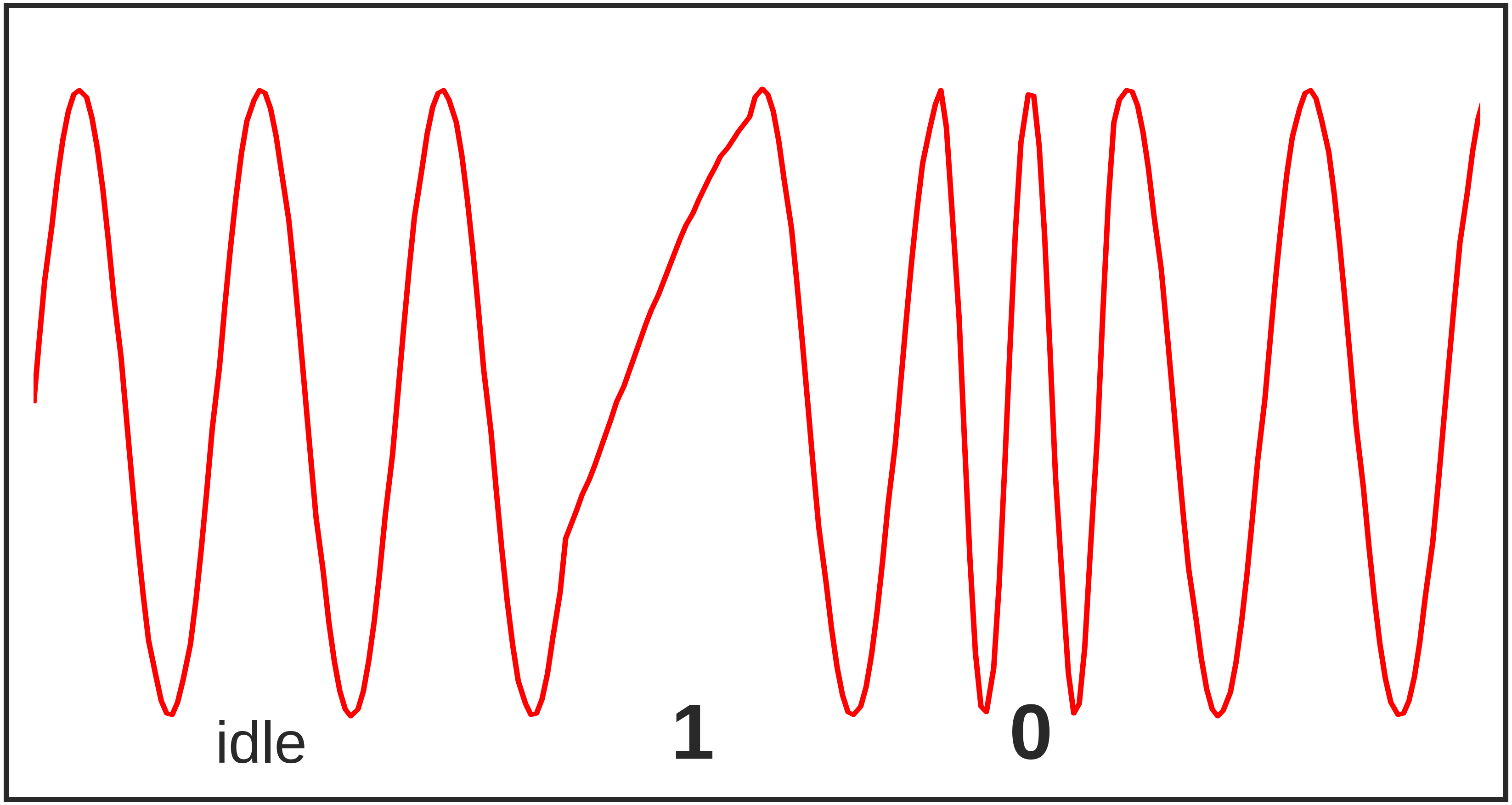
Binary Voltage Encodings

- Problem with binary voltage (square wave) encodings
  - Wide frequency range required, implying
    - Significant dispersion
    - Uneven attenuation
  - Prefer to use narrow frequency band (carrier frequency)
- Types of modulation
  - Amplitude (AM)
  - Frequency (FM)
  - Phase/phase shift
  - Combinations of these

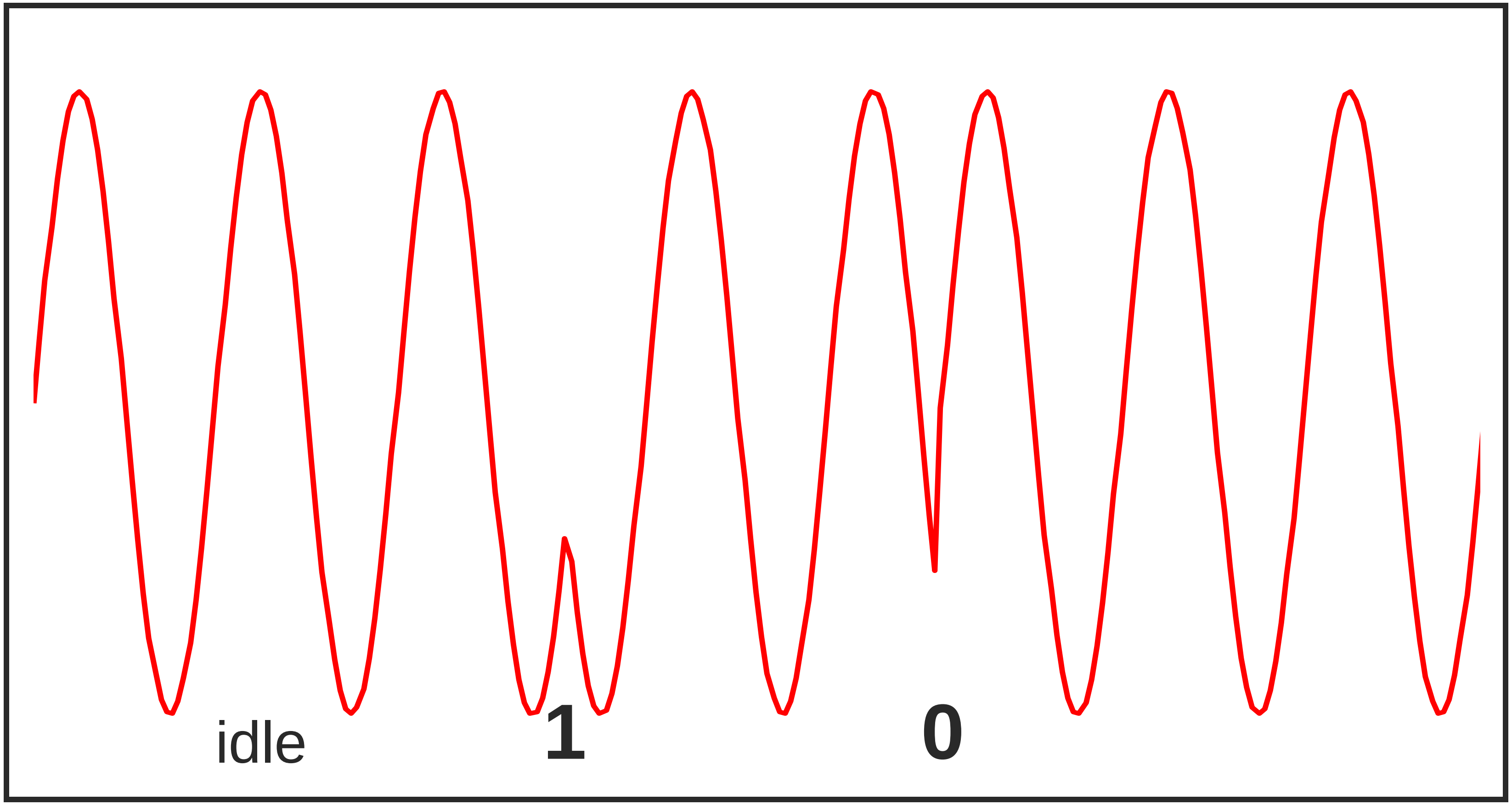
Amplitude Modulation



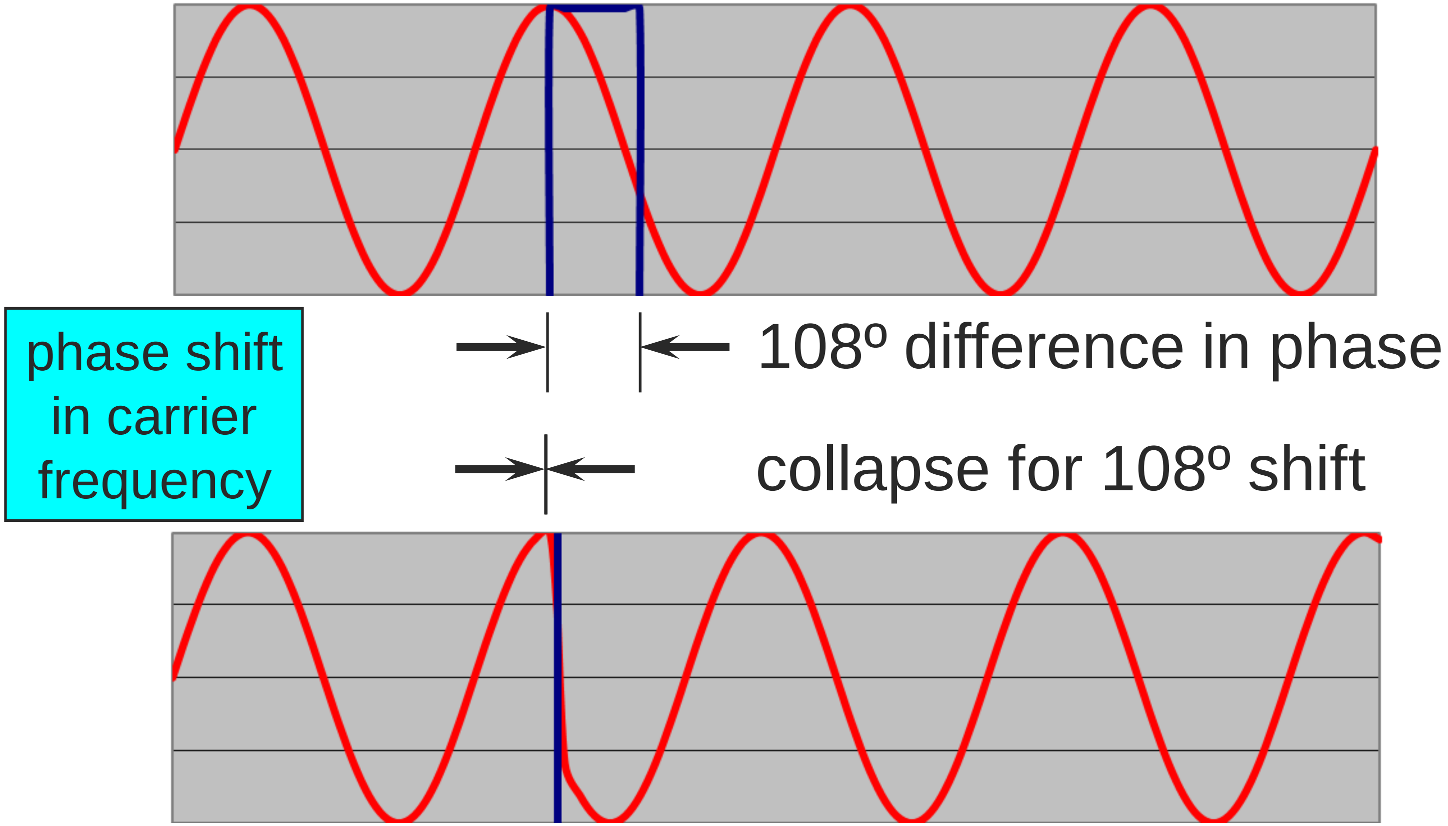
Frequency Modulation



Phase Modulation



Phase Modulation



phase shift in carrier frequency

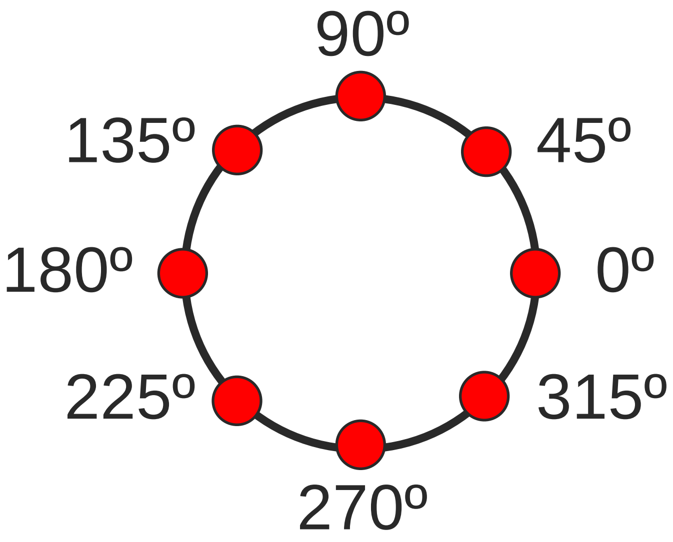
108º difference in phase

collapse for 108º shift

Phase Modulation Algorithm

- Send carrier frequency for one period
  - Perform phase shift
  - Shift value encodes symbol
    - Value in range [0, 360º)
    - Multiple values for multiple symbols
    - Represent as circle

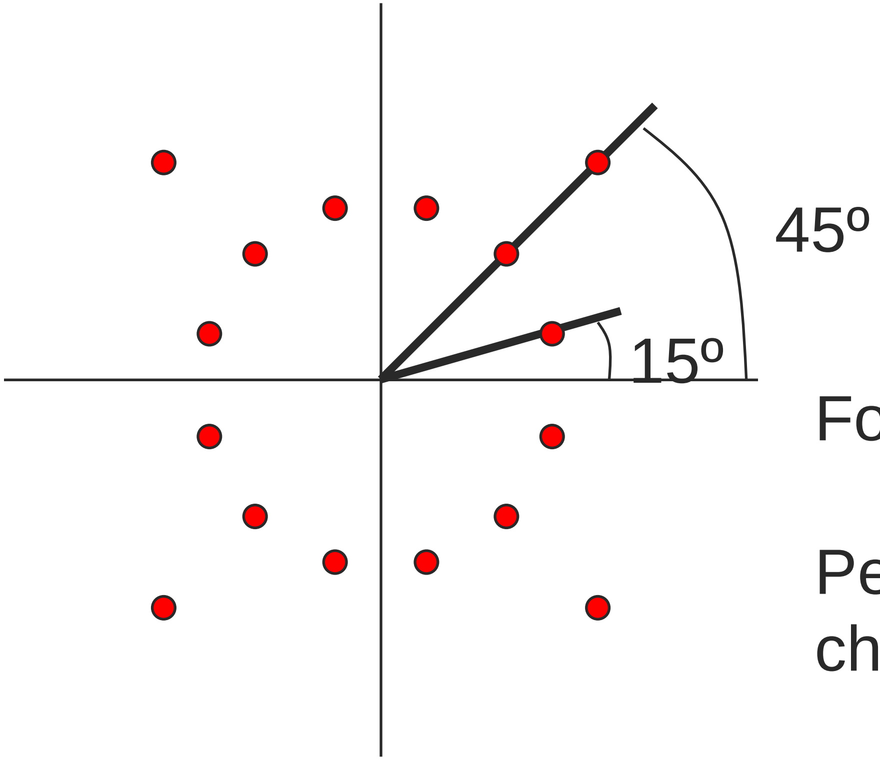
8-symbol example



V.32 9600 bps

- Communication between modems
- Analog phone line
- Uses a combination of amplitude and phase modulation
  - Known as Quadrature Amplitude Modulation (QAM)
- Sends one of 16 signals each clock cycle

Constellation Pattern for V.32 QAM



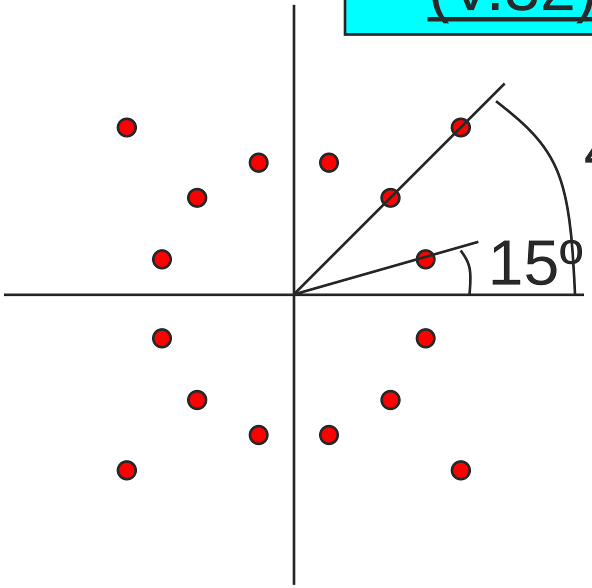
For a given symbol:

Perform phase shift and change to new amplitude

Quadrature Amplitude Modulation (QAM)

- Same algorithm as phase modulation
- Can also change signal amplitude
- 2-dimensional representation
  - Angle is phase shift
  - Radial distance is new amplitude

16-symbol example (V.32)



Comments on V.32

- V.32 transmits at 2400 baud
  - *i.e.*, 2,400 symbols per second
- Each symbol contains $\log_2 16 = 4$ bits
  - Data rate is thus 4 x 2400 = 9600 bps
- Points in constellation diagram
  - Chosen to maximize error detection
  - Process called trellis coding

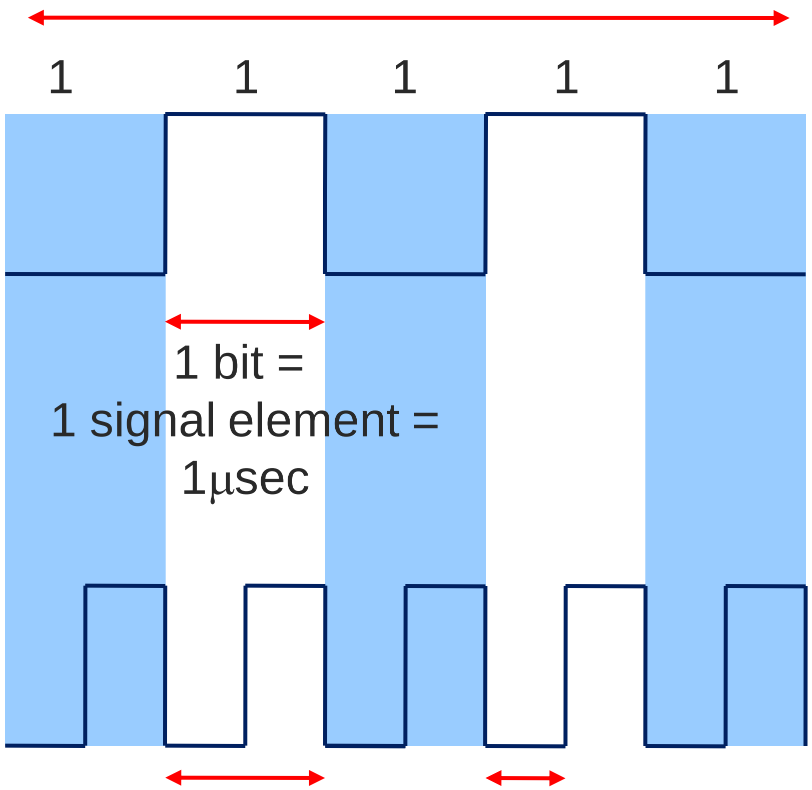
Generalizing the Examples

- What limits baud rate?
- What data rate can a channel sustain?
- How is data rate related to bandwidth?
- How does noise affect these bounds?
- What else can limit maximum data rate?

What Limits Baud Rate?

- Baud rate
  - Typically limited by electrical signaling properties
- Changing voltages takes time
  - No matter how small the voltage or how short the wire
- Electronics
  - Slow compared to optics
- Note
  - Baud rate can be as high as twice the frequency (bandwidth) of communication
  - One cycle can contain two symbols

Modulation Rate



5 bits = 5μsec

1 1 1 1 1

NRZI

A stream of binary 1s at 1 Mbps

1 bit =
1 signal element =
1μsec

Manchester

1 bit =
1μsec

1 signal element =
0.5μsec

Data Rate (R)
= bits/sec
= 1 Mbps for both

Modulation Rate
= Baud Rate
= Rate at which signal elements are generated
= R (NRZI)
= 2R (Manchester)

What Data Rate can a Channel Sustain? How is Data Rate Related to Bandwidth?

- Transmitting N distinct signals over a noiseless channel with bandwidth B, we can achieve at most a data rate of

$$2B \log_2 N$$

- Nyquist's Sampling Theorem (H. Nyquist, 1920's)
  - Sampling rate = 2B
  - A higher sampling rate is pointless because higher frequency signals have been filtered out

Noiseless Capacity

- Example 1: sampling rate of a phone line
  - B = 4000 Hz
  - 2B = 8000 samples/sec.
    - sample every 125 microseconds!!
- Example 2: noiseless capacity
  - D = 2400 baud {note D = 2H}
  - V = each pulse encodes 16 levels
  - C = 2H $\log_2$ (V) = D x $\log_2$ (V)

    = 2400 x 4 = 9600 bps.

What else (Besides Noise) can Limit Maximum Data Rate?

- Transitions between symbols
  - Introduce high-frequency components into the transmitted signal
  - Such components cannot be recovered (by Nyquist’s Theorem), and some information is lost
- Examples
  - Phase modulation
    - Single frequency (with different phases) for each symbol
    - Transitions can require very high frequencies

How does Noise affect these Bounds?

- In-band (thermal, not high-frequency) noise
  - Blurs the symbols, reducing the number of symbols that can be reliably distinguished.
- Claude Shannon (1948)
  - Extended Nyquist's work to channels with additive white Gaussian noise (a good model for thermal noise)

$$\text{channel capacity } C = B \log_2 (1 + S/N)$$

B is the channel bandwidth

S/N is the ratio between

the average signal power and

the average in-band noise power

Noisy Capacity

- Telephone channel
  - 3400 Hz at 40 dB SNR
  - $C = B \log_2 (1+S/N)$ b/s
  - $S/N = 40$ dB

    $40 = 10 \log_{10} (S/N)$
  - $4 = \log_{10} (S/N)$

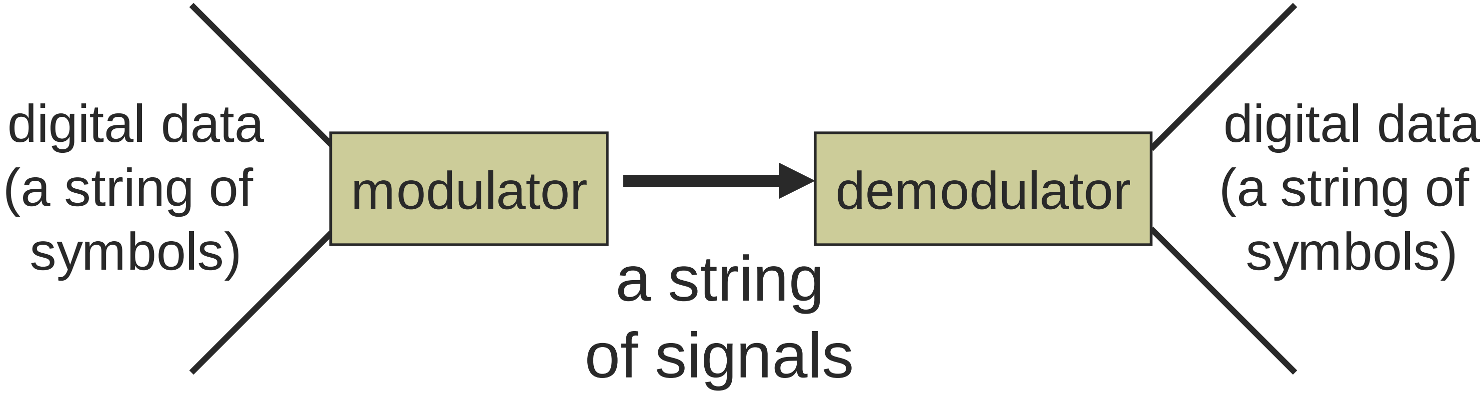
    $S/N = 10{,}000$
  - $C = 3400 \log_2 (10001) = 44.8$ kbps

S/N (dB) = $10 \log_{10}$ S/R

Summary of Encoding

- Problems
    - Attenuation, dispersion, noise
- Digital transmission allows periodic regeneration
- Variety of binary voltage encodings
    - High frequency components limit to short range
    - More voltage levels provide higher data rate
- Carrier frequency and modulation
    - Amplitude, frequency, phase, and combinations
    - Quadrature amplitude modulation: amplitude and phase, many signals
- Nyquist (noiseless) and Shannon (noisy) limits on data rates

Framing


digital data (a string of symbols)

modulator

demodulator

digital data (a string of symbols)

a string of signals

- Encoding translates symbols to signals
- Framing demarcates units of transfer
  - Separates continuous stream of bits into frames
  - Marks start and end of each frame

Framing

- Demarcates units of transfer
- Goal
  - Enable nodes to exchange blocks of data
- Challenge
  - How can we determine exactly what set of bits constitute a frame?
  - How do we determine the beginning and end of a frame?

Framing

- Synchronization recovery
  - Breaks up continuous streams of unframed bytes
  - Recall RS-232 start and stop bits
- Link multiplexing
  - Multiple hosts on shared medium
  - Simplifies multiplexing of logical channels
- Efficient error detection
  - Per-frame error checking and recovery

Framing

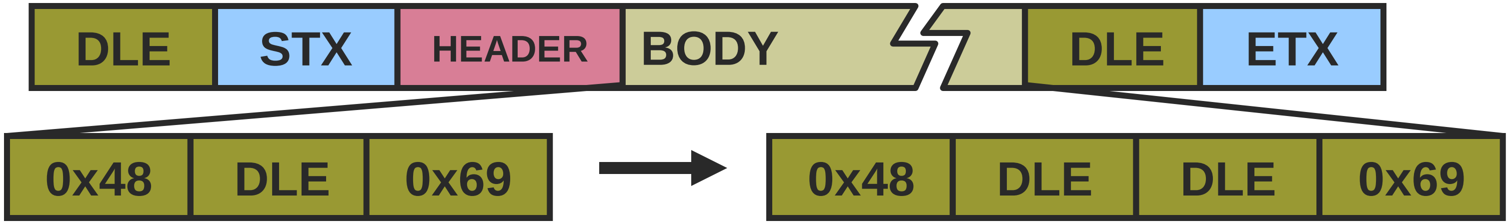
- Approaches
  - Sentinel (like C strings)
  - Length-based (like Pascal strings)
  - Clock based
- Characteristics
  - Bit- or byte-oriented
  - Fixed or variable length
  - Data-dependent or data-independent length

Sentinel-Based Framing

- End of Frame
    - Marked with a special byte or bit pattern
        - Frame length is data-dependent
    - Challenge
        - Frame marker may exist in data
        - Requires stuffing
- Examples
    - BISYNC, HDLC, PPP, IEEE 802.4 (token bus)

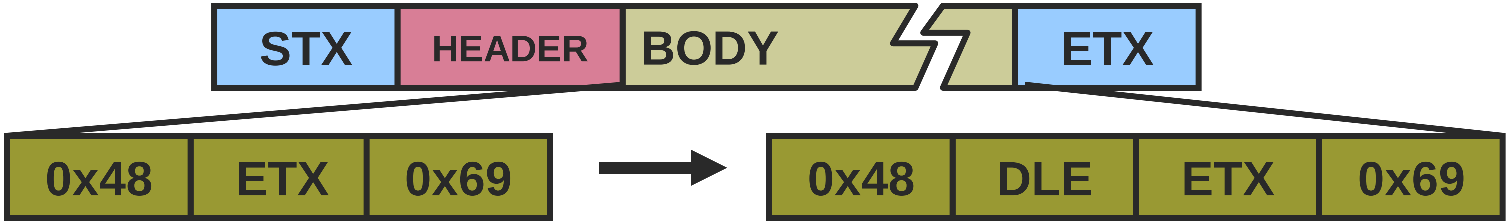
ARPANET IMP-IMP

- Interface Message processors (IMPs)
  - Packet switching nodes in the original ARPANET
  - Byte oriented, Variable length, Data dependent
  - Frame marker bytes
    - STX/ETX start of text/end of text
    - DLE data link escape
  - Byte Stuffing
    - DLE byte in data sent as two DLE bytes back-to-back



BISYNC

- BInary SYNchronous Communication
  - Developed by IBM in late 1960's
  - Byte oriented, Variable length, Data dependent
  - Frame marker bytes:
    - STX/ETX start of text/end of text
    - DLE data link escape
  - Byte Stuffing
    - ETX/DLE bytes in data prefixed with DLE's



High-Level Data Link Control Protocol (HDLC)

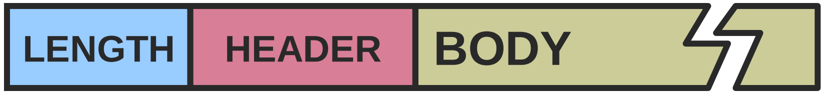
- Bit oriented, Variable length, Data-dependent
- Frame Marker
  - 01111110
- Bit Stuffing
  - Insert 0 after pattern 011111 in data
  - Example
    - 01111110 end of frame
    - 01111111 error! lose one or two frames

IEEE 802.4 (token bus)

- Alternative to Ethernet (802.3) with fairer arbitration
- End of frame marked by encoding violation,
  - i.e., physical signal not used by valid data symbol
  - Recall Manchester encoding
    - low-high means “0”
    - high-low means “1”
    - low-low and high-high are invalid
- IEEE 802.4
  - byte-oriented, variable-length, data-independent
- Another example
  - Fiber Distributed Data Interface (FDDI) uses 4B/5B
- Technique also applicable to bit-oriented framing

Length-Based Framing

- End of frame
  - Calculated from length sent at start of frame
  - Challenge
    - Corrupt length markers
- Examples
  - DECNET's DDCMP
    - Byte-oriented, variable-length
  - RS-232 framing
    - Bit-oriented, implicit fixed-length

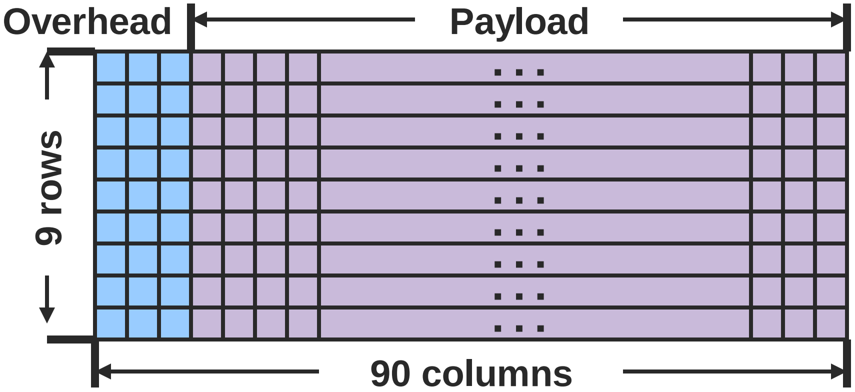


Clock-Based Framing

- Continuous stream of fixed-length frames
  - Clocks must remain synchronized
- STS-1 frames - 125μs long
  - No bit or byte stuffing
- Example
  - Synchronous Optical Network (SONET)
- Problems
  - Frame synchronization
  - Clock synchronization

SONET

- Frames (all STS formats) are 125 μsec long
  - Ex: STS-1 – 51.84 Mbps = 90 bytes
- Frame Synchronization
  - 2-byte synchronization pattern at start of each frame

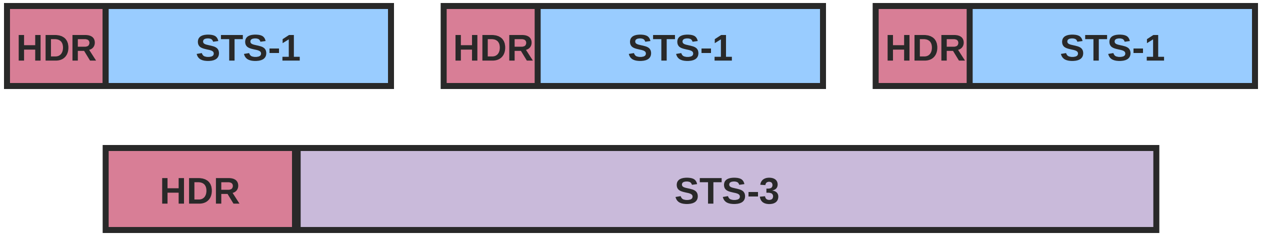


SONET: Challenges

- How to recover frame synchronization
  - Synchronization pattern unlikely to occur in data
    - Wait until pattern appears in same place repeatedly
- How to maintain clock synchronization
  - NRZ encoding
    - Data scrambled (XOR'd) with 127-bit pattern
    - Creates transitions
    - Also reduces chance of finding false sync. pattern

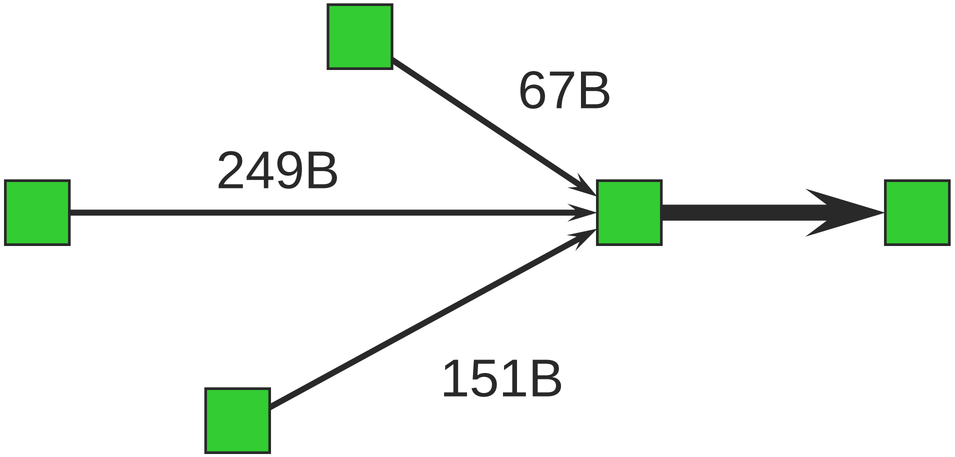
SONET

- A single SONET frame may contain multiple smaller SONET frames
- Bytes from multiple SONET frames are interleaved to ensure pacing



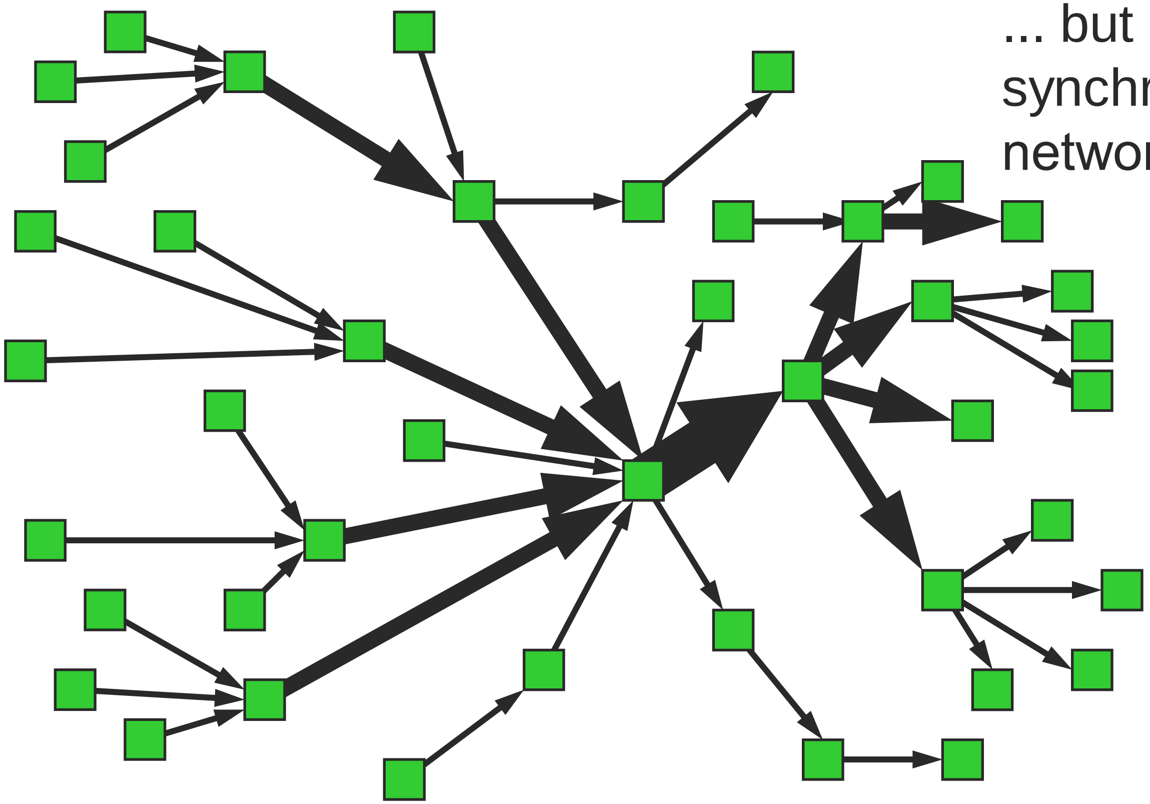
SONET

- STS-1 merged bytewise round-robin into STS-3
- Unmerged (single-source) format called STS-3c
- Problem: simultaneous synchronization of many distributed clocks



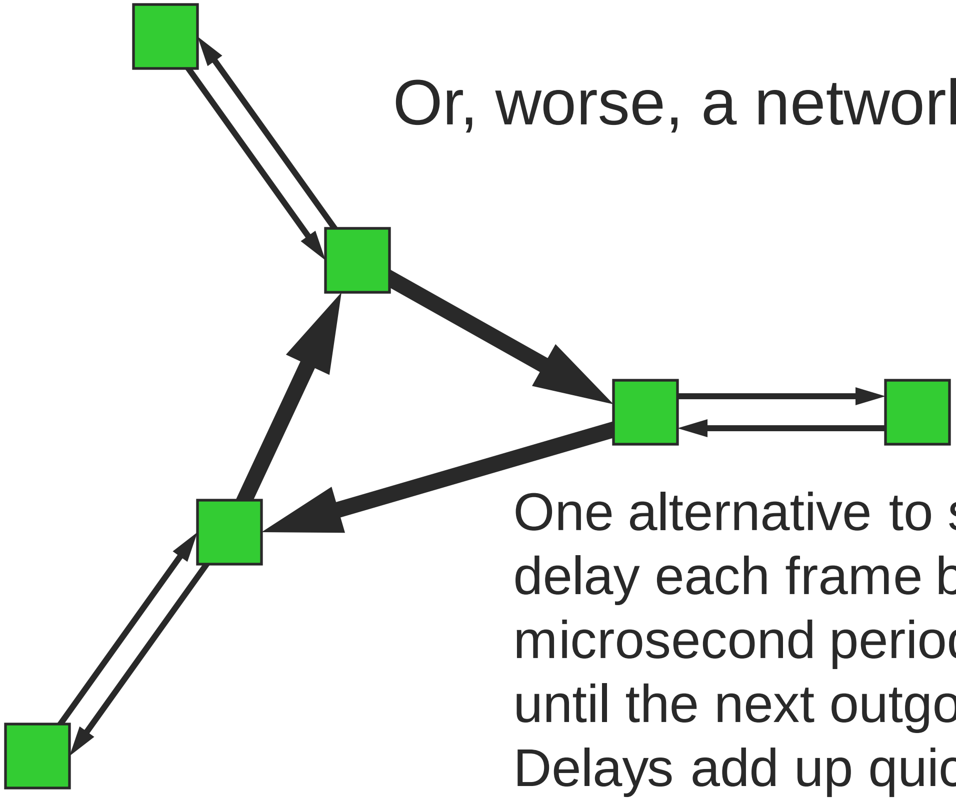
not too difficult to synchronize clocks such that first byte of all incoming flows arrives just before sending first 3 bytes of outgoing flow

SONET



... but now try to synchronize this network’s clocks

SONET

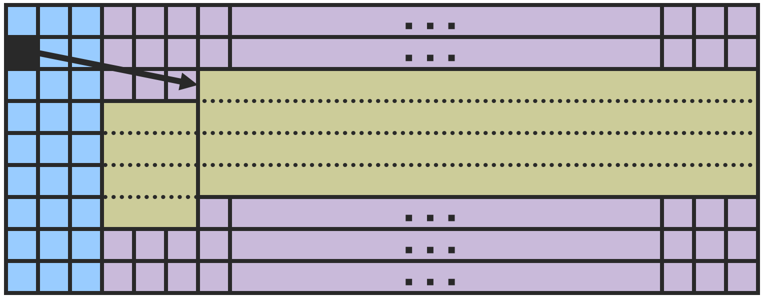


Or, worse, a network with cycles.

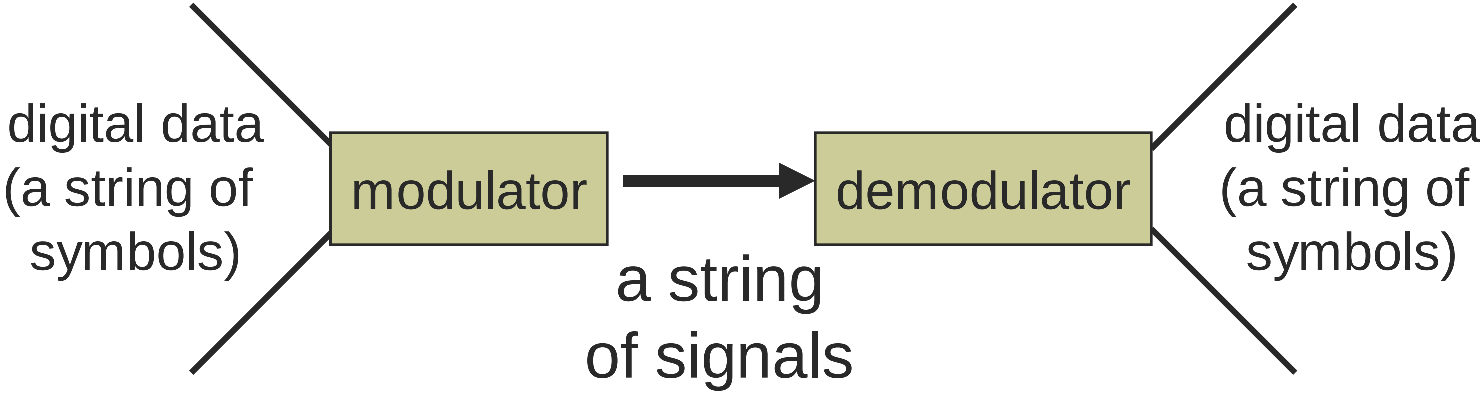
One alternative to synchronization is to delay each frame by some fraction of a 125 microsecond period at each switch (i.e., until the next outgoing frame starts). Delays add up quickly...

SONET

- Problem
  - Clock synchronization across multiple machines
- Solution
  - Allow payload to float across frame boundaries
  - Part of overhead specifies first byte of payload



Error Detection



digital data (a string of symbols)

modulator

demodulator

a string of signals

digital data (a string of symbols)

- Encoding translates symbols to signals
- Framing demarcates units of transfer
- Error detection validates correctness of each frame

Error Detection

- Idea
    - Add redundant information that can be used to determine if errors have been introduced, and potentially fix them
- Errors checked at many levels
    - Demodulation of signals into symbols (analog)
    - Bit error detection/correction (digital)—our main focus
        - Within network adapter (CRC check)
        - Within IP layer (IP checksum)
        - Possibly within application as well

Error Detection

- Analog Errors
  - Example of signal distortion
- Hamming distance
  - Parity and voting
  - Hamming codes
- Error bits or error bursts?
- Digital error detection
  - Two-dimensional parity
  - Checksums
  - Cyclic Redundancy Check (CRC)

Analog Errors

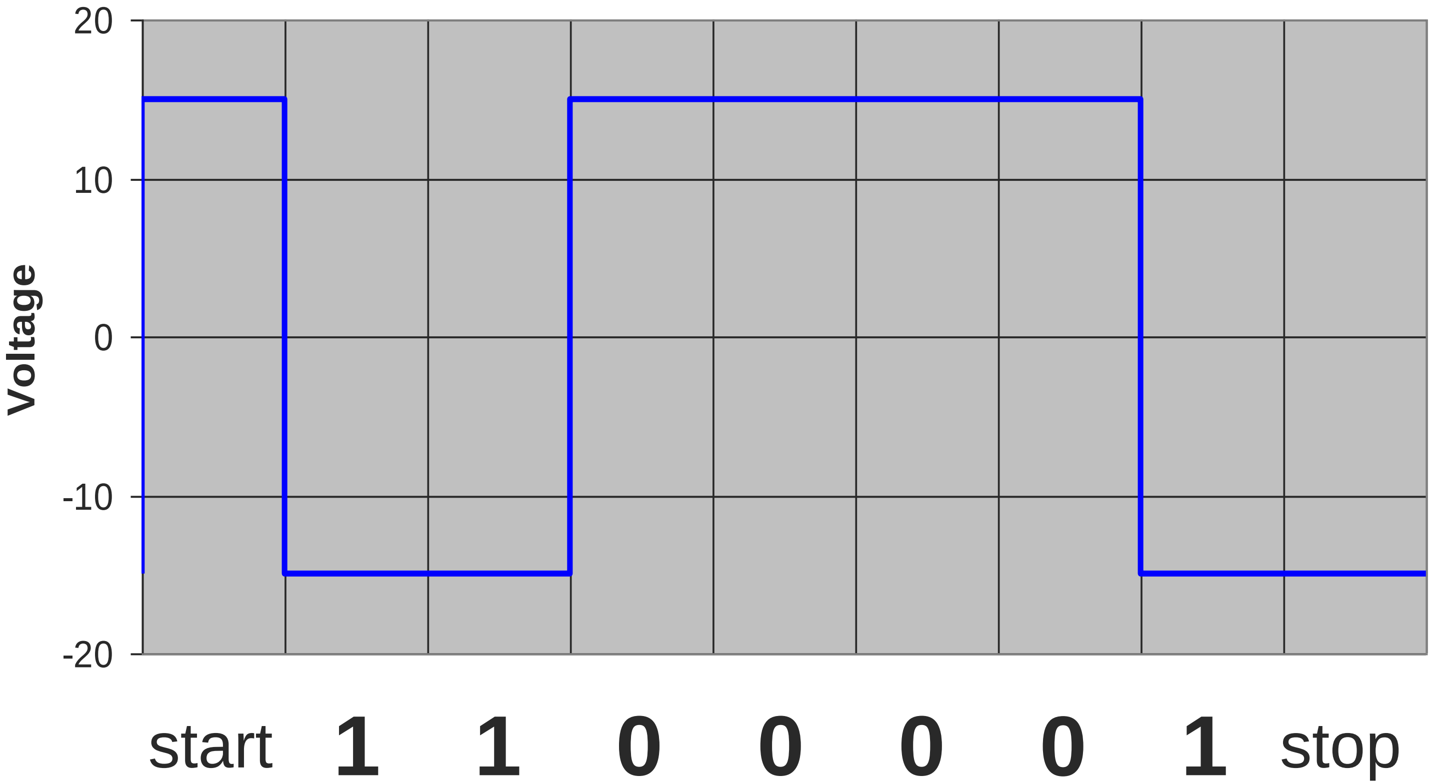
- Consider RS-232 encoding of character ‘Q’
- Assume idle wire (-15V) before and after signal
- Calculate frequency distribution of signal A(f) using a Fourier transform:

$$A(f) = \int_{-\infty}^{\infty} x(t)[\cos(2\pi f t) + i\sin(2\pi f t)]\,dt$$

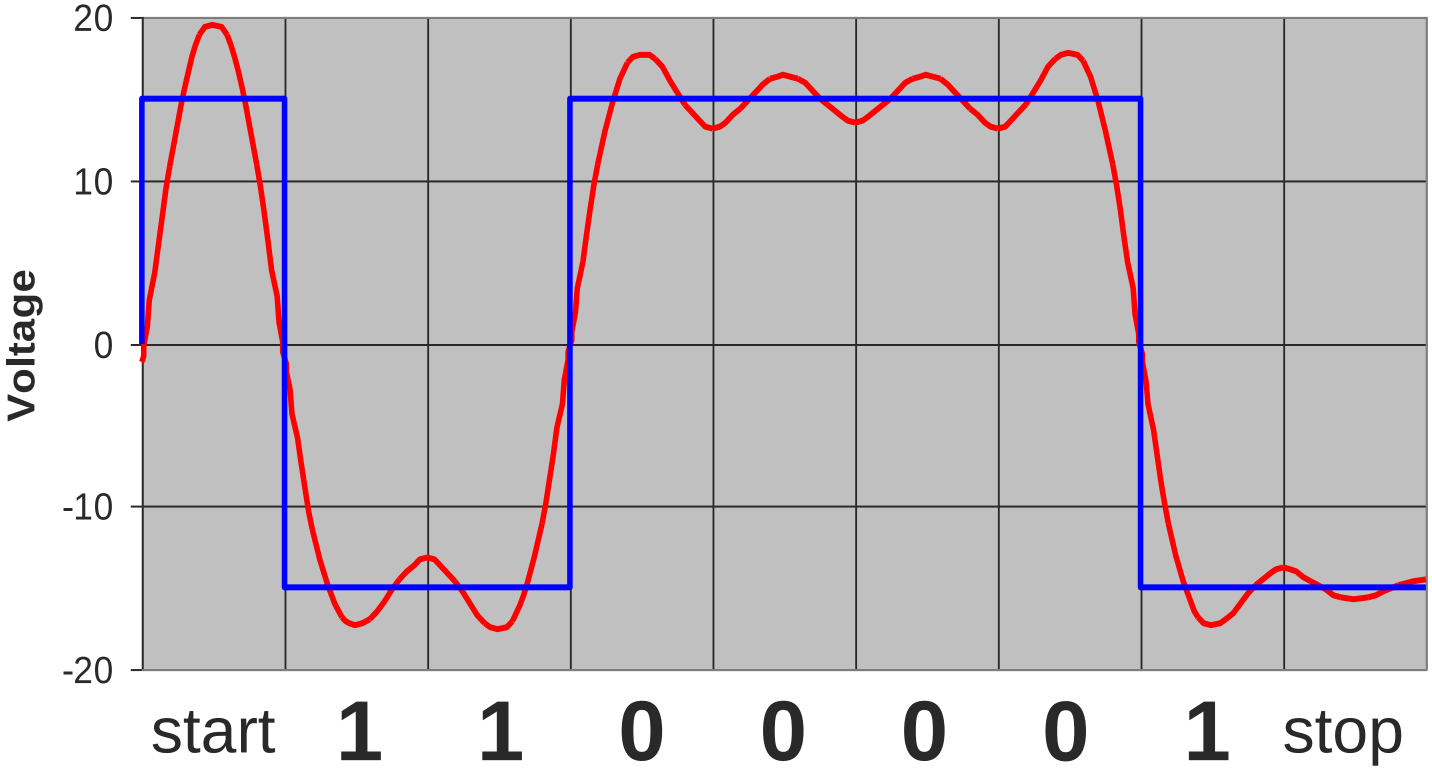
$$x(t) = \int_{-\infty}^{\infty} A(f)[\cos(2\pi f t) - i\sin(2\pi f t)]\,df$$

- Apply low-pass filter (drop high frequency components)
- Calculate signal using inverse Fourier transform above

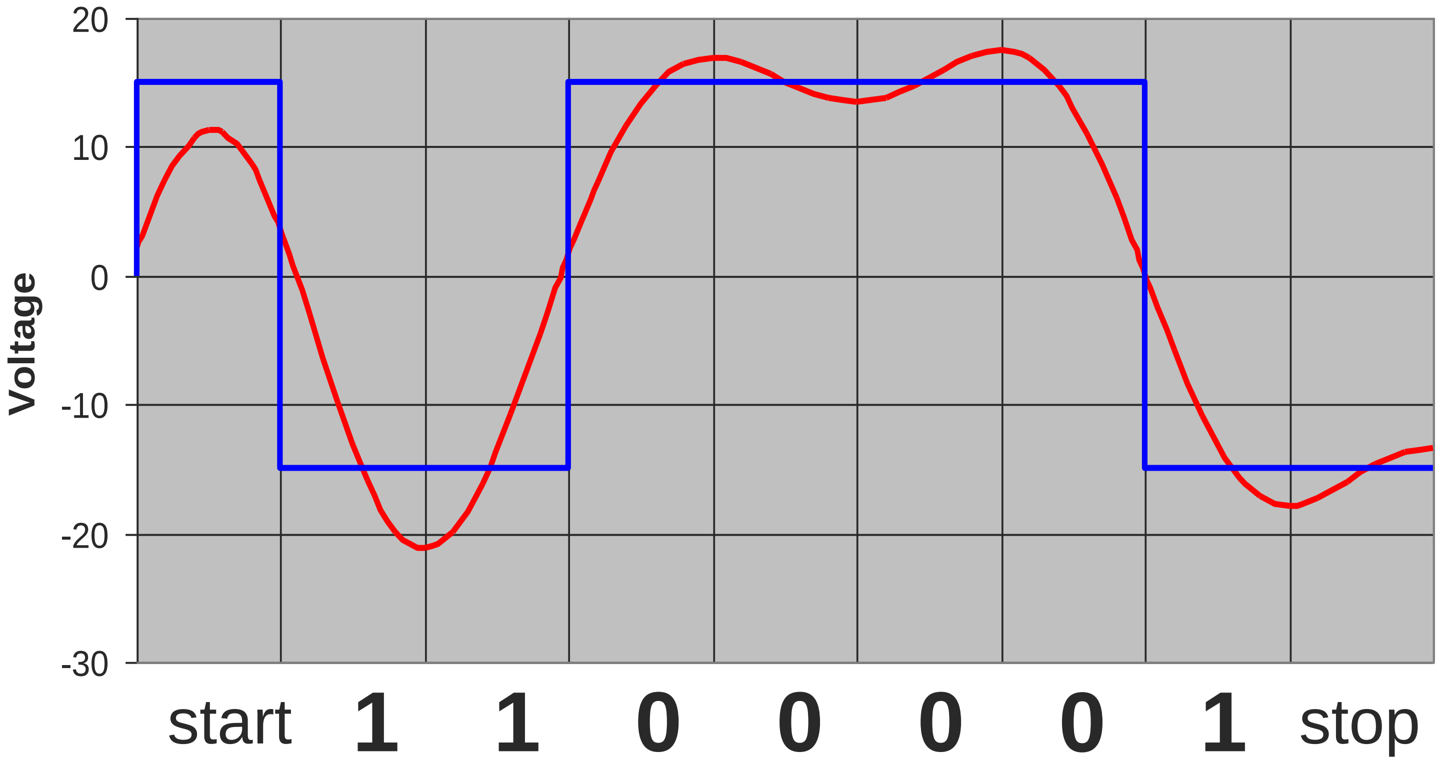
RS-232 Encoding of 'Q'



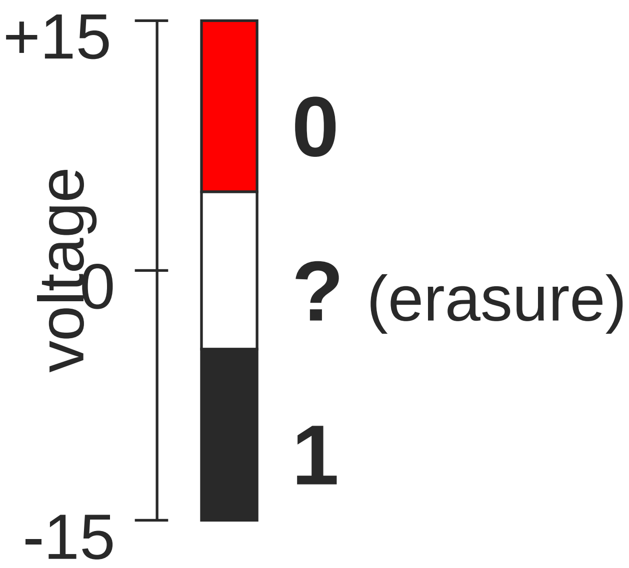
Limited-Frequency Signal Response (bandwidth = baud rate)



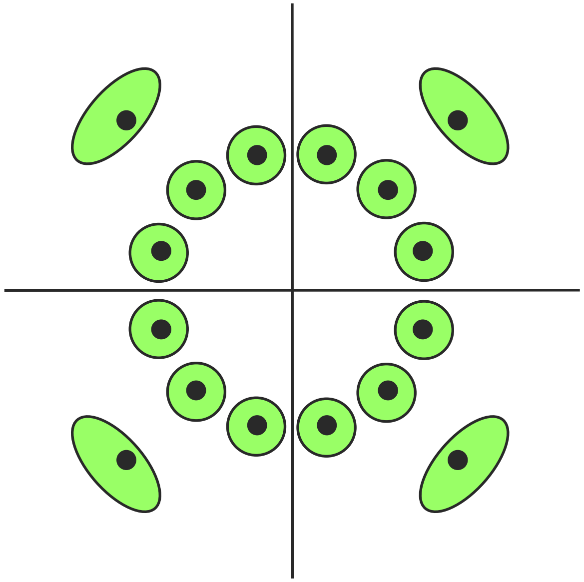
Limited-Frequency Signal Response (bandwidth = baud rate/2)



Symbols



possible binary voltage encoding symbol neighborhoods and erasure region



possible QAM symbol neighborhoods in green; all other space results in erasure

Symbols

- Inputs to digital level
  - valid symbols
  - erasures
- Hamming distance
  - Definition
  - 1-bit error-detection with parity
  - 1-bit error-correction with voting
  - 2-bit erasure-correction with voting
  - Hamming codes (1-bit error correction)

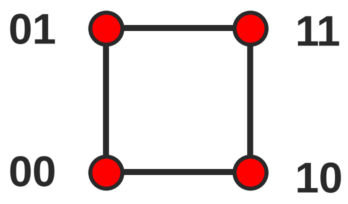
Hamming Distance

- The Hamming distance between two code words is the minimum number of bit flips to move from one to the other
  - Example:
  - 00101 and 00010
  - Hamming distance of 3

Parity

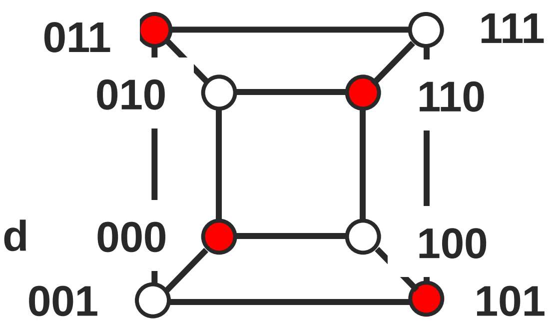
- 1-bit error detection with parity
  - Add an extra bit to a code to ensure an even (odd) number of 1s
  - Every code word has an even (odd) number of 1s

Valid code words



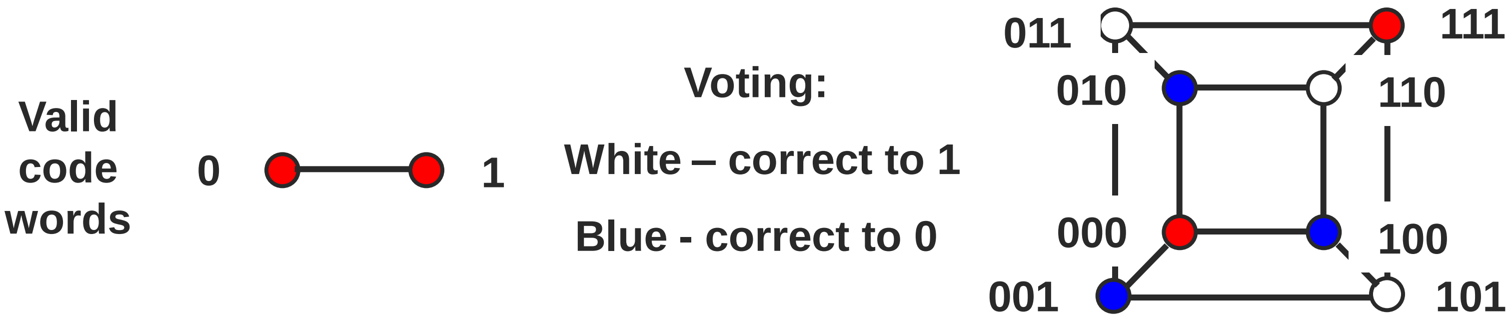
Parity Encoding:

White – invalid (error)



Voting

- 1-bit error correction with voting
  - Every codeword is transmitted n times



Valid code words

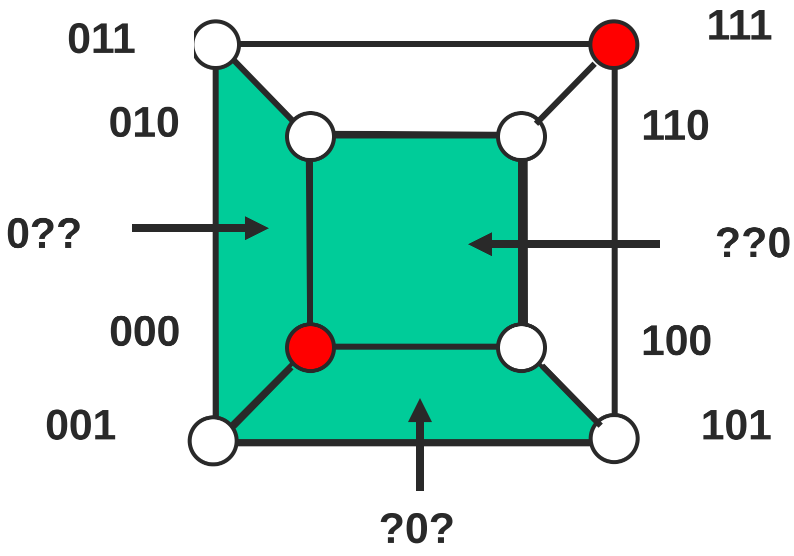
Voting:

White – correct to 1

Blue - correct to 0

2-bit Erasure Correction with Voting

- Every code word is copied 3 times



2-erasure planes in green
remaining bit not ambiguous

cannot correct 1-error and 1-erasure

Minimum Hamming Distance

- The minimum Hamming distance of a code is the minimum distance over all pairs of codewords
  - Minimum Hamming Distance for parity
    - 2
  - Minimum Hamming Distance for voting
    - 3

Coverage

- N-bit error detection
  - No code word changed into another code word
  - Requires Hamming distance of N+1
- N-bit error correction
  - N-bit neighborhood: all codewords within N bit flips
  - No overlap between N-bit neighborhoods
  - Requires hamming distance of 2N+1

Hamming Codes

- Construction for 1-bit error-correcting codes
- Minimal number of check bits required
- Construction
  - number bits from 1 upward
  - powers of 2 are check bits
  - all others are data bits
  - Check bit j is XOR of all bits k such that (j AND k) = j
- Example:
  - 4 bits of data, 3 check bits

Hamming Codes

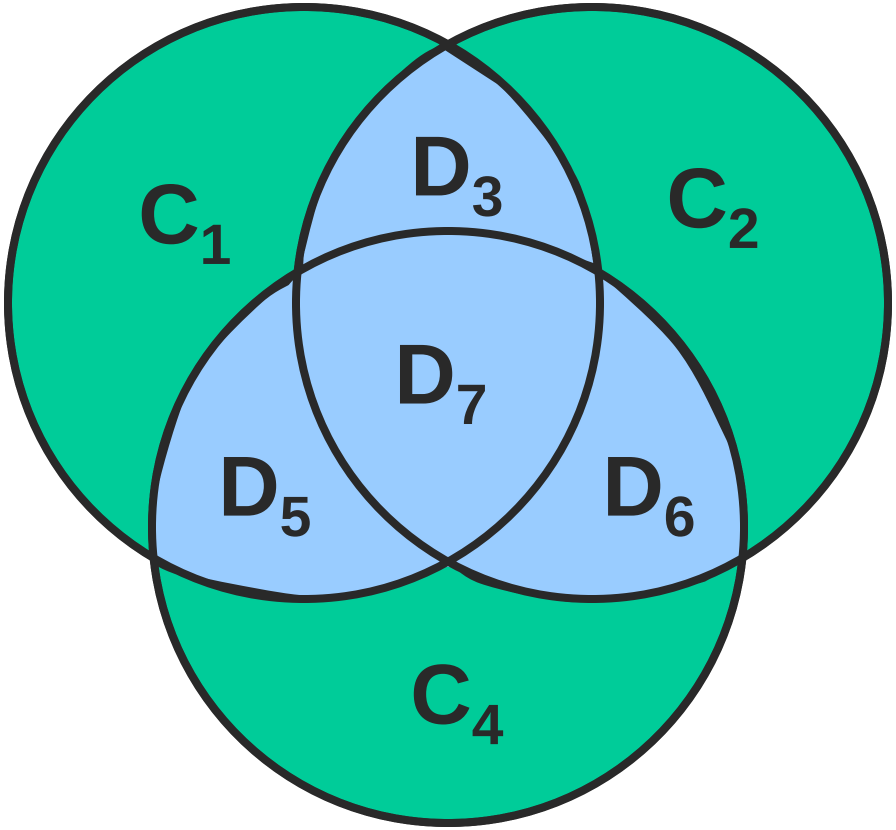
1	2	3	4	5	6	7
C_1	C_2	D_3	C_4	D_5	D_6	D_7

C1 = D3 XOR D5 XOR D7

C2 = D3 XOR D6 XOR D7

C4 = D5 XOR D6 XOR D7

Hamming Codes



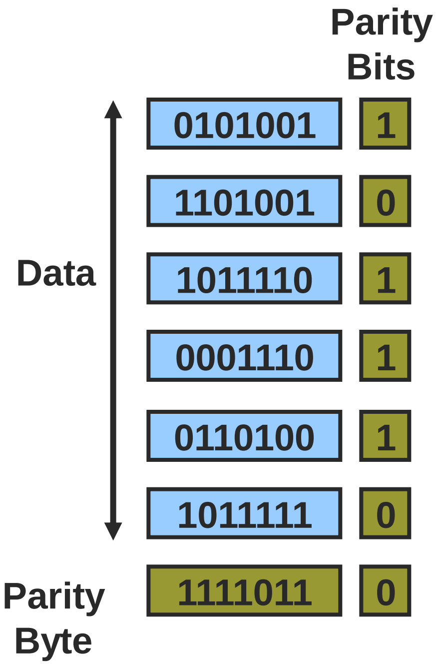
Error Bits or Bursts?

- Common model of errors
  - Probability of error per bit
  - Error in each bit independent of others
  - Value of incorrect bit independent of others
- Burst model
  - Probability of back-to-back bit errors
  - Error probability dependent on adjacent bits
  - Value of errors may have structure
- Why assume bursts?
  - Appropriate for some media (e.g., radio)
  - Faster signaling rate enhances such phenomena

Digital Error Detection Techniques

- Two-dimensional parity
  - Detects up to 3-bit errors
  - Good for burst errors
- IP checksum
  - Simple addition
  - Simple in software
  - Used as backup to CRC
- Cyclic Redundancy Check (CRC)
  - Powerful mathematics
  - Tricky in software, simple in hardware
  - Used in network adapter

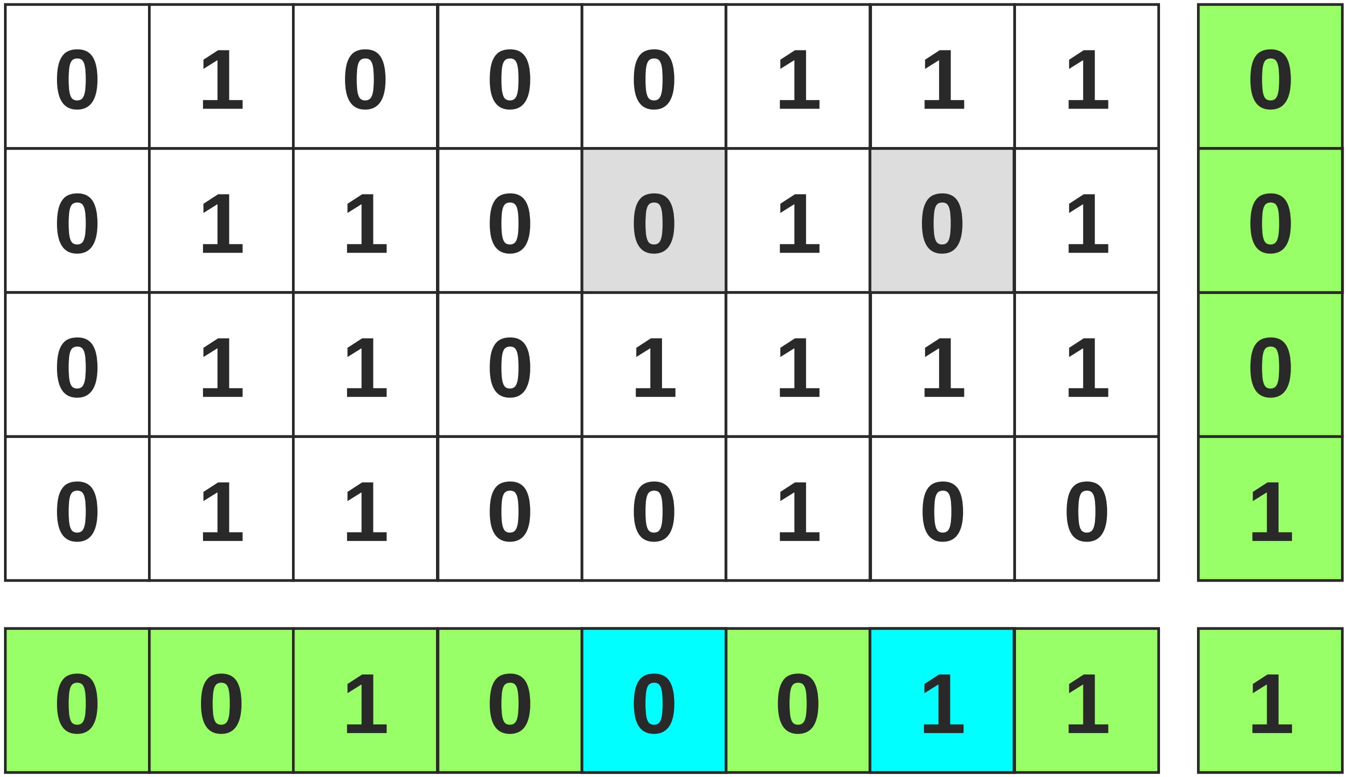
Two-Dimensional Parity



		Parity Bits
Data	0101001	1
	1101001	0
	1011110	1
	0001110	1
	0110100	1
	1011111	0
Parity Byte	1111011	0

- Use 1-dimensional parity
  - Add one bit to a 7-bit code to ensure an even/odd number of 1s
- Add 2nd dimension
  - Add an extra byte to frame
    - Bits are set to ensure even/odd number of 1s in that position across all bytes in frame
- Comments
  - Catches all 1-, 2- and 3-bit and most 4-bit errors

Two-Dimensional Parity



0	1	0	0	0	1	1	1	0
0	1	1	0	0	1	0	1	0
0	1	1	0	1	1	1	1	0
0	1	1	0	0	1	0	0	1
0	0	1	0	0	0	1	1	1

Internet Checksum

- Idea
    - Add up all the words
    - Transmit the sum
- Internet Checksum
    - Use 1’s complement addition on 16bit codewords
    - Example
        - Codewords: -5 -3
        - 1’s complement binary: 1010 1100
        - 1’s complement sum 1000
- Comments
    - Small number of redundant bits
    - Easy to implement
    - Not very robust

IP Checksum

```
u_short cksum(u_short *buf, int count) {
    register u_long sum = 0;
    while (count--) {
        sum += *buf++;
        if (sum & 0xFFFF0000) {
        /* carry occurred, so wrap around */
                sum &= 0xFFFF;
                sum++;
        }
    }
    return ~(sum & 0xFFFF);
}
```

Cyclic Redundancy Check (CRC)

- Goal
  - Maximize protection, Minimize extra bits
- Idea
  - Add k bits of redundant data to an n-bit message
  - N-bit message is represented as a n-degree polynomial with each bit in the message being the corresponding coefficient in the polynomial
  - Example
    - Message = 10011010
    - Polynomial

$$= \mathbf{1} * x^7 + \mathbf{0} * x^6 + \mathbf{0} * x^5 + \mathbf{1} * x^4 + \mathbf{1} * x^3 + \mathbf{0} * x^2 + \mathbf{1} * x + \mathbf{0}$$

$$= x^7 + x^4 + x^3 + x$$

CRC

- Select a divisor polynomial C(x) with degree k
  - Example with k = 3:
    - $C(x) = x^3 + x^2 + 1$
    - Represented as 1101
- Transmit a polynomial P(x) that is evenly divisible by C(x)
  - P(x) = M(x) + k bits

Properties of Polynomial Arithmetic

- Divisor
    - Any polynomial B(x) can be divided by a polynomial C(x) if B(x) is of the same or higher degree than C(x)
- Remainder
    - The remainder obtained when B(x) is divided by C(x) is obtained by subtracting C(x) from B(x)
- Subtraction
    - To subtract C(x) from B(x), simply perform an XOR on each pair of matching coefficients

CRC - Sender

- Given
  - M(x) = 10011010 = $x^7 + x^4 + x^3 + x$
  - C(x) = 1101 = $x^3 + x^2 + 1$
- Steps
  - T(x) = M(x) by x^k (zero extending)
  - Find remainder, R(x), from T(x)/C(x)
  - P(x) = T(x) – R(x) ⇒ M(x) followed by R(x)
- Example
  - T(x) = 10011010000
  - R(x) = 101
  - P(x) = 10011010101

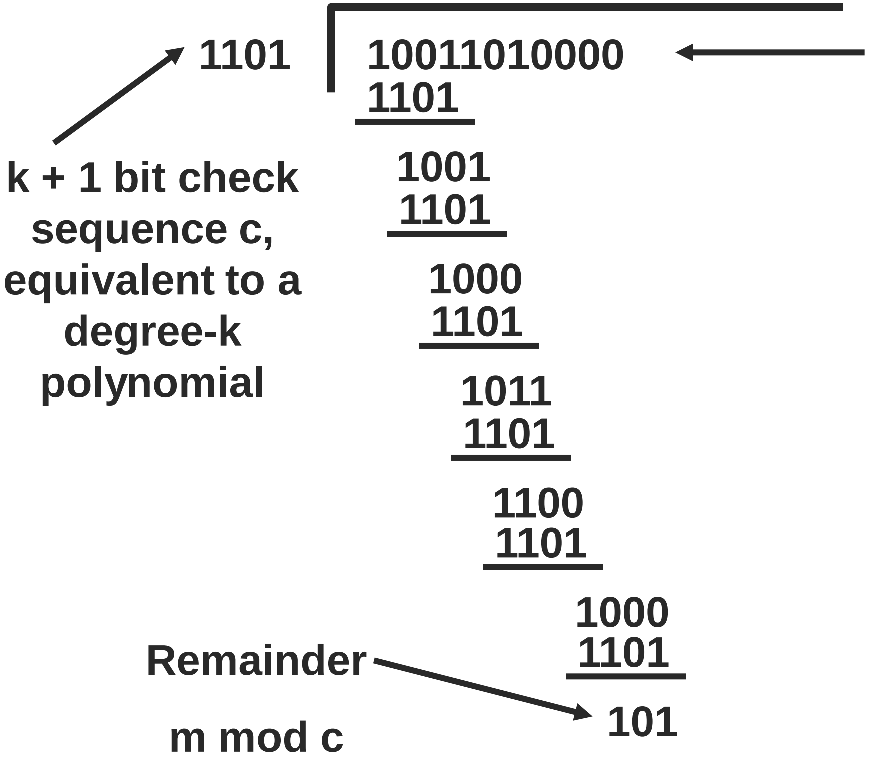
CRC - Receiver

- Receive Polynomial P(x) + E(x)
  - E(x) represents errors
  - E(x) = 0, implies no errors
- Divide (P(x) + E(x)) by C(x)
  - If result = 0, either
    - No errors (E(x) = 0, and P(x) is evenly divisible by C(x))
    - (P(x) + E(x)) is exactly divisible by C(x), error will not be detected

CRC – Example Encoding

C(x) = $x^3 + x^2 + 1$ = 1101 Generator

M(x) = $x^7 + x^4 + x^3 + x$ = 10011010 Message



```
1101 | 10011010000
       1101
        1001
        1101
         1000
         1101
          1011
          1101
           1100
           1101
              1000
              1101
               101
```

Message plus k zeros

k + 1 bit check sequence c, equivalent to a degree-k polynomial

Remainder

m mod c

Result:

Transmit message followed by remainder:

10011010101

CRC – Example Decoding – No Errors

C(x) = $x^3 + x^2 + 1$ **= 1101** **Generator**

P(x) = $x^{10} + x^7 + x^6 + x^4 + x^2 + 1$ **= 10011010101** **Received Message**

```
1101 | 10011010101
       1101
       ----
        1001
        1101
        ----
         1000
         1101
         ----
          1011
          1101
          ----
           1100
           1101
           ----
             1101
             1101
             ----
                0
```

k + 1 bit check sequence c, equivalent to a degree-k polynomial

Received message, no errors

Remainder

m mod c

Result:

CRC test is passed

CRC – Example Decoding – with Errors

C(x) = $x^3 + x^2 + 1$ **= 1101** **Generator**

P(x) = $x^{10} + x^7 + x^5 + x^4 + x^2 + 1$ **= 10010110101** **Received Message**

```
1101 | 10010110101
       1101
       ----
        1000
        1101
        ----
         1011
         1101
         ----
          1101
          1101
          ----
           0101
```

Received message

Two bit errors

k + 1 bit check sequence c, equivalent to a degree-k polynomial

Remainder

m mod c

Result:

CRC test failed

CRC Error Detection

- Properties
  - Characterize error as E(x)
  - Error detected unless C(x) divides E(x)
    - (*i.e.*, E(x) is a multiple of C(x))

Example of Polynomial Multiplication

- Multiply
  - 1101 by 10110
  - $x^3 + x^2 + 1$ by $x^4 + x^2 + x$

```
         1011
        10110
       ------

        1101
      1101
    1101
   ----------
00011111110
```

This is a multiple of c, so that if errors occur according to this sequence, the CRC test would be passed

On Polynomial Arithmetic

- Polynomial arithmetic
  - A fancy way to think about addition with no carries.
  - Helps in the determination of a good choice of C(x)
  - A non-zero vector is not detected if and only if the error polynomial E(x) is a multiple of C(x)
- Implication
  - Suppose C(x) has the property that C(1) = 0 (i.e. (x + 1) is a factor of C(x))
  - If E(x) corresponds to an undetected error pattern, then it must be that E(1) = 0
  - Therefore, any error pattern with an odd number of error bits is detected

CRC Error Detection

- What errors can we detect?
  - All single-bit errors, if x^k and x^0 have non-zero coefficients
  - All double-bit errors, if $C(x)$ has at least three terms
  - All odd bit errors, if $C(x)$ contains the factor $(x + 1)$
  - Any bursts of length $< k$, if $C(x)$ includes a constant term
  - Most bursts of length $\geq k$

Common Polynomials for C(x)

CRC	C(x)
CRC-8	$x^8 + x^2 + x^1 + 1$
CRC-10	$x^{10} + x^9 + x^5 + x^4 + x^1 + 1$
CRC-12	$x^{12} + x^{11} + x^3 + x^2 + x^1 + 1$
CRC-16	$x^{16} + x^{15} + x^2 + 1$
CRC-CCITT	$x^{16} + x^{12} + x^5 + 1$
CRC-32	$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x^1 + 1$

Error Detection vs. Error Correction

- Detection
  - Pro: Overhead only on messages with errors
  - Con: Cost in bandwidth and latency for retransmissions
- Correction
  - Pro: Quick recovery
  - Con: Overhead on all messages
- What should we use?
  - Correction if retransmission is too expensive
  - Correction if probability of errors is high