CSCI 1680
Physical Layer, Link Layer I

Nick DeMarinis

Based partly on lecture notes by Rodrigo Fonseca, David Mazières, Phil Levis, John Jannotti

Administrivia

- Snowcast: Milestone due today (ish)
  - Make sure you follow our submission format
  - So long as you pass the tests locally or with reference, you're fine
- Snowcast full submission: due Monday 9/25
- HW1: details soon

Last call for override codes
If you emailed me yesterday, I will respond after class

Roadmap

- One thing on sockets
- Physical layer key points
- Inherent properties of *real* networks

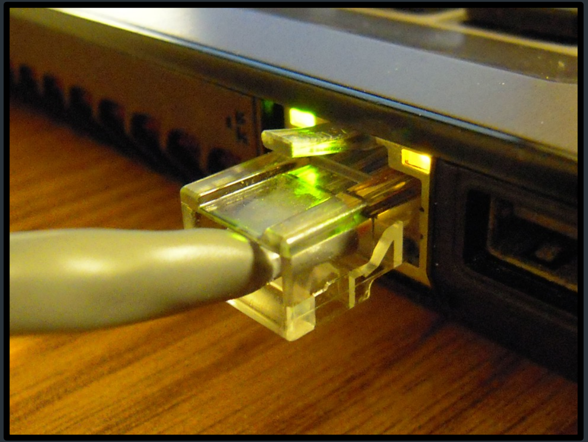
Layers, Services, Protocols

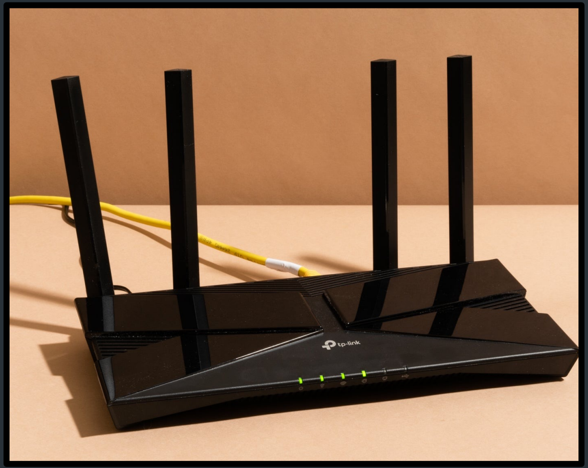
Application	Service: user-facing application. Application-defined messages
Transport	Service: multiplexing applications Reliable byte stream to other node (TCP), Unreliable datagram (UDP)
Network	Service: move packets to any other node in the network IP: Unreliable, best-effort service model
Link	Service: move frames to other node across link. May add reliability, medium access control
Physical	Service: move bits to other node across link

Physical Layer (Layer 1)

Specifies three things:

- Physical medium
- Signaling/modulation
- Encoding

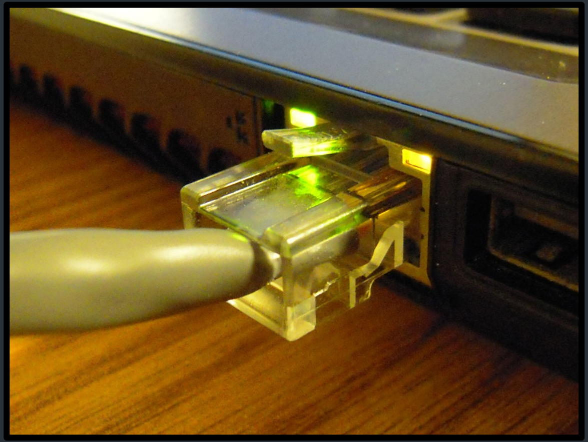


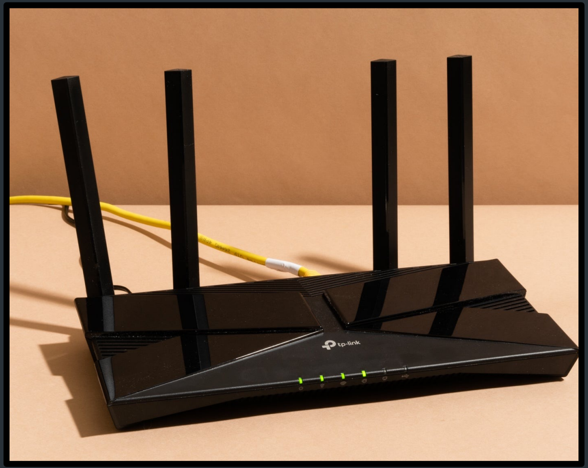


Physical Layer (Layer 1)

Specifies three things:

- Physical medium: cable, fiber, wireless frequency
- Signaling/modulation: how to transmit/receive
- *Encoding:* how to get meaningful data





Why should we care?

This is the line between electrical engineering and computer science

Helpful to understand challenges involved
=> How design/limitations affect our systems

Also: Learn important principles we'll use elsewhere

The main idea

- Send/receive data over a *medium* (copper wire, fiber, radio frequency)
- Sender *encodes* message using some format, sends "over the wire"
- Receiver *decodes (or recovers) message at the other end*

How does this work?

What can go wrong?

- Noise
- Sharing channel: interference from other devices
- Physical distance (attenuation)
- Energy usage
- Security

=> Every medium has its own characteristics, and problems

Key points

- All media have fixed <u>bandwidth</u> => fixed “space” to transmit information
- Sending data takes time! => latency
- All media have (some) errors => how to deal with them?

Bandwidth

Bandwidth

- **Bandwidth** – frequencies that a channel propagates well
  – Signals consist of many frequency components

- Creates a fixed "space" in which data can be transmitted
  => Wires: defined by physical properties
  ⇒ Wireless: frequency ranges are regulated

Upper bound on *throughput*: amount of data we can send per time (bits per second)

UNITED STATES FREQUENCY ALLOCATIONS

THE RADIO SPECTRUM

RADIO SERVICES COLOR LEGEND

- AERONAUTICAL MOBILE
- AERONAUTICAL MOBILE SATELLITE
- AERONAUTICAL RADIONAVIGATION
- AMATEUR
- AMATEUR SATELLITE
- BROADCASTING
- BROADCASTING SATELLITE
- EARTH EXPLORATION SATELLITE
- FIXED
- FIXED SATELLITE
- INTER-SATELLITE
- LAND MOBILE
- LAND MOBILE SATELLITE
- MARITIME MOBILE
- MARITIME MOBILE SATELLITE
- MARITIME RADIONAVIGATION
- METEOROLOGICAL
- METEOROLOGICAL SATELLITE
- MOBILE
- MOBILE SATELLITE
- RADIO ASTRONOMY
- RADIODETERMINATION SATELLITE
- RADIOLOCATION
- RADIOLOCATION SATELLITE
- RADIONAVIGATION
- RADIONAVIGATION SATELLITE
- SPACE OPERATION
- SPACE RESEARCH
- STANDARD FREQUENCY AND TIME SIGNAL
- STANDARD FREQUENCY AND TIME SIGNAL SATELLITE

ACTIVITY CODE

- FEDERAL EXCLUSIVE
- FEDERAL/NON-FEDERAL SHARED
- NON-FEDERAL EXCLUSIVE

ALLOCATION USAGE DESIGNATION

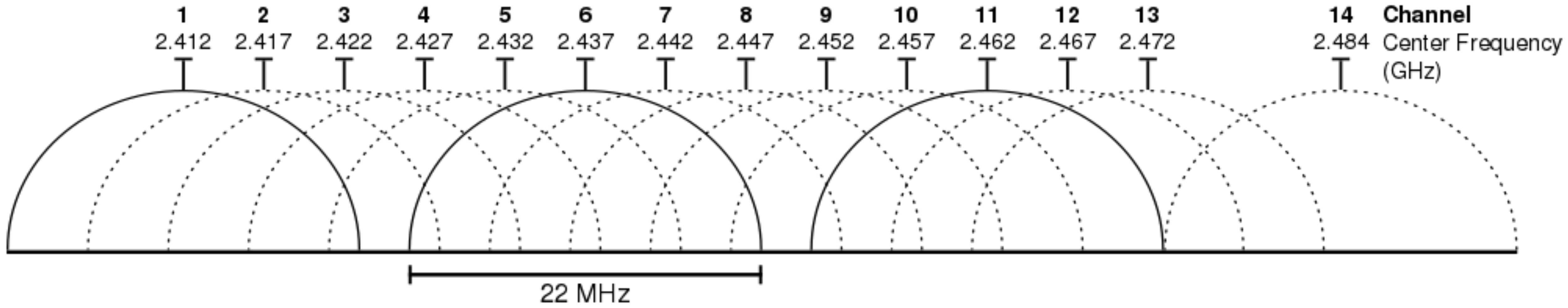
SERVICE	EXAMPLE	DESCRIPTION
Primary	FIXED	Capital Letters
Secondary	Mobile	1st Capital with lower case letters

This chart is a graphic single-point-in-time portrayal of the Table of Frequency Allocations used by the FCC and NTIA. As such, it may not completely reflect all aspects, i.e. footnotes and recent changes made to the Table of Frequency Allocations. Therefore, for complete information, users should consult the Table to determine the current status of U.S. allocations.

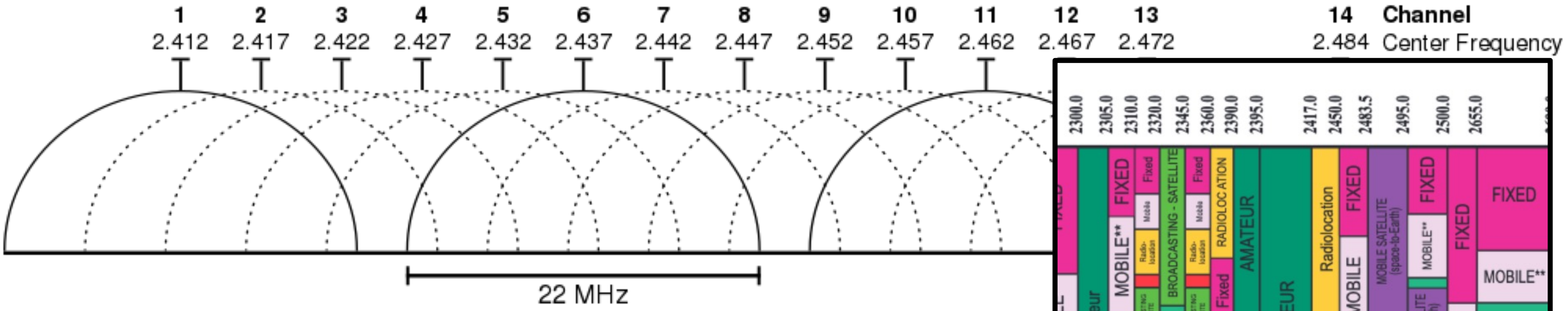
U.S. DEPARTMENT OF COMMERCE
National Telecommunications and Information Administration
Office of Spectrum Management

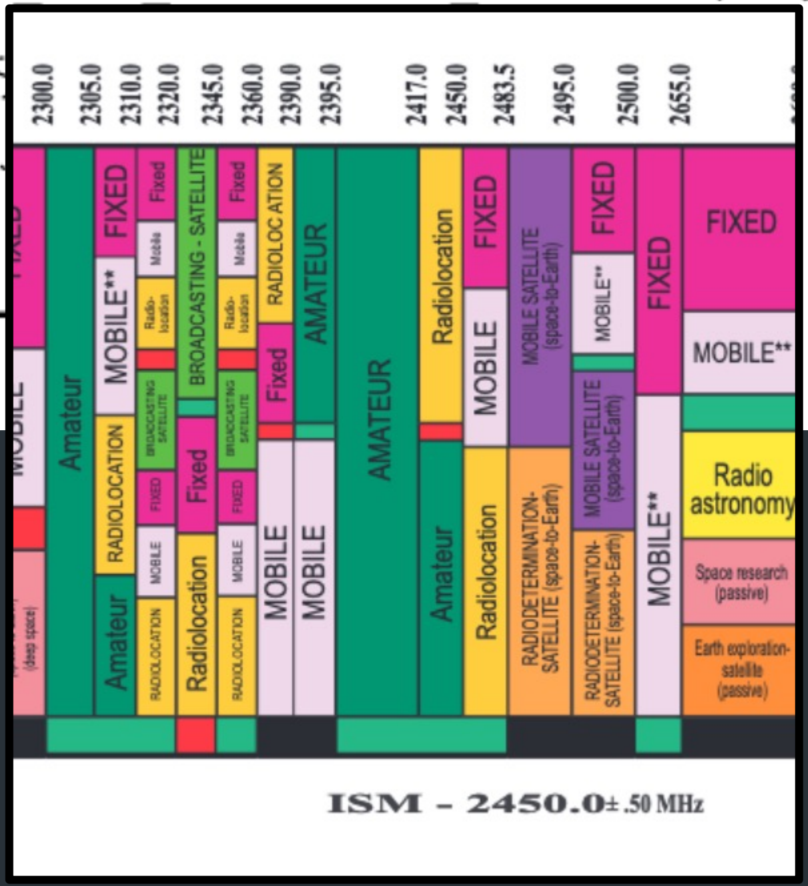
JANUARY 2016

Early IEEE 802.11 (Wifi) channel bandwidth



Early IEEE 802.11 (Wifi) channel bandwidth





Wi-Fi generations

V · T · E

Generation	IEEE standard	Adopted	Maximum link rate (Mbit/s)	Radio frequency (GHz)
Wi-Fi 7	802.11be	(2024)	1376 to 46120	2.4/5/6
Wi-Fi 6E	802.11ax	2020	574 to 9608[41]	6[42]
Wi-Fi 6		2019		2.4/5
Wi-Fi 5	802.11ac	2014	433 to 6933	5[43]
Wi-Fi 4	802.11n	2008	72 to 600	2.4/5
*(Wi-Fi 3)**	802.11g	2003	6 to 54	2.4
	802.11a	1999		5
*(Wi-Fi 2)**	802.11b	1999	1 to 11	2.4
*(Wi-Fi 1)**	802.11	1997	1 to 2	2.4

*(Wi-Fi 1, 2, and 3 are by retroactive inference) [44][45][46][47][48]

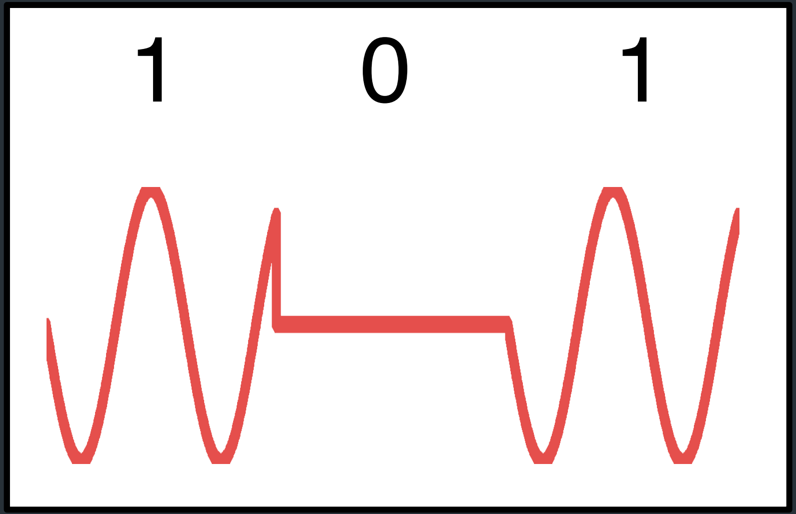
How to actually send stuff?

Modulation: how to vary a signal in order to transmit information

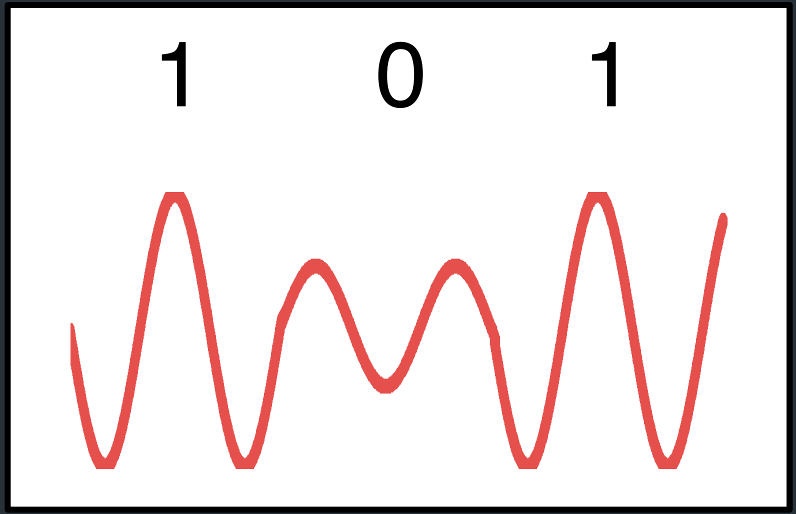
One way: Use Carriers

Start with a *carrier frequency*, *modulate* it to encode data:

OOK: On-Off Keying



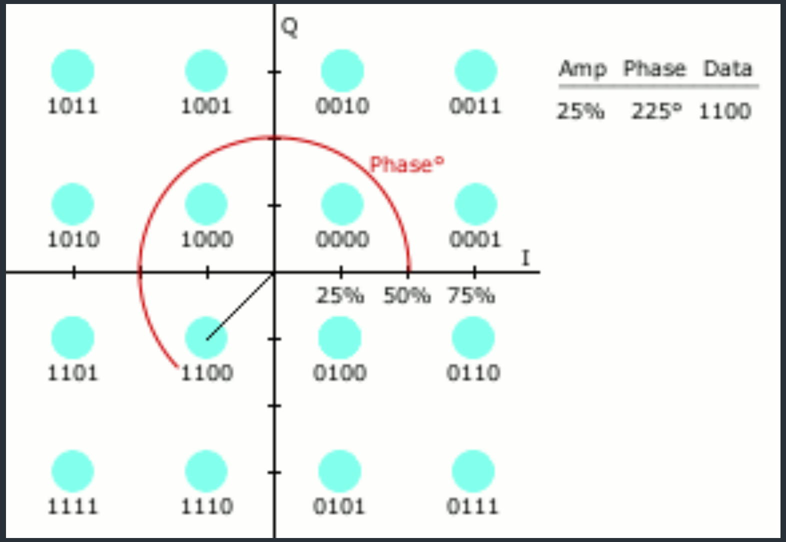
ASK: Amplitude Shift Keying



This can get more complex…

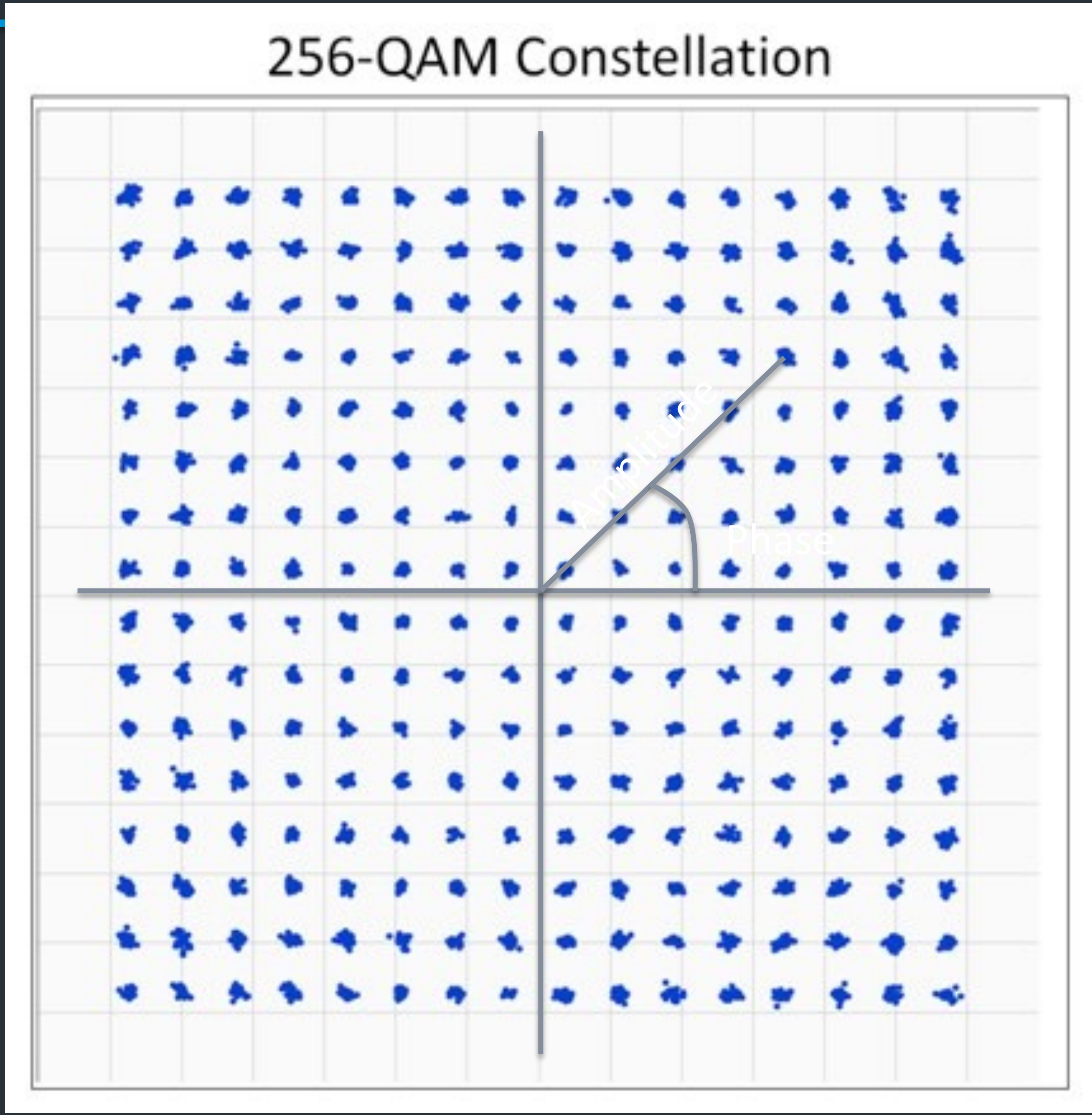
Lots of engineering you can do

- Multiple carriers/frequencies
- Adjust amplitude, phase
- Clever ways to avoid errors
- …



A good animation on Wikipedia

Example: Quadrature Amplitude Modulation (QAM)



Modulation schemes in action

- https://www.youtube.com/watch?v=vvr9AMWEU-c

Sounds great, right?

- Problem: noise limits the number of modulation levels (M)

Shannon's Law: $C = B \log_2(1 + S/N)$

- C: channel capacity in bits/second
- B : bandwidth in Hz
- S, N: average signal, noise power

=> For any medium, need to design encodings based on bandwidth, noise characteristics

Medium	Bandwidth	Throughput
Dialup	8 kHz	56 Kbit/s
Early Wifi (802.11g)	20 MHz	54 Mbit/s
Modern Wifi (802.11ax)	20-40 MHz	Up to 9 Gbps
Ethernet	62.5 MHz (1Gbps version)	1Gbit/s (common) Up to 100Gbps
3G cellular	Depends on carrier	2 Mbit/s
5G cellular	Depends on carrier	> 1 GBps

=> Does this mean wifi is the best?

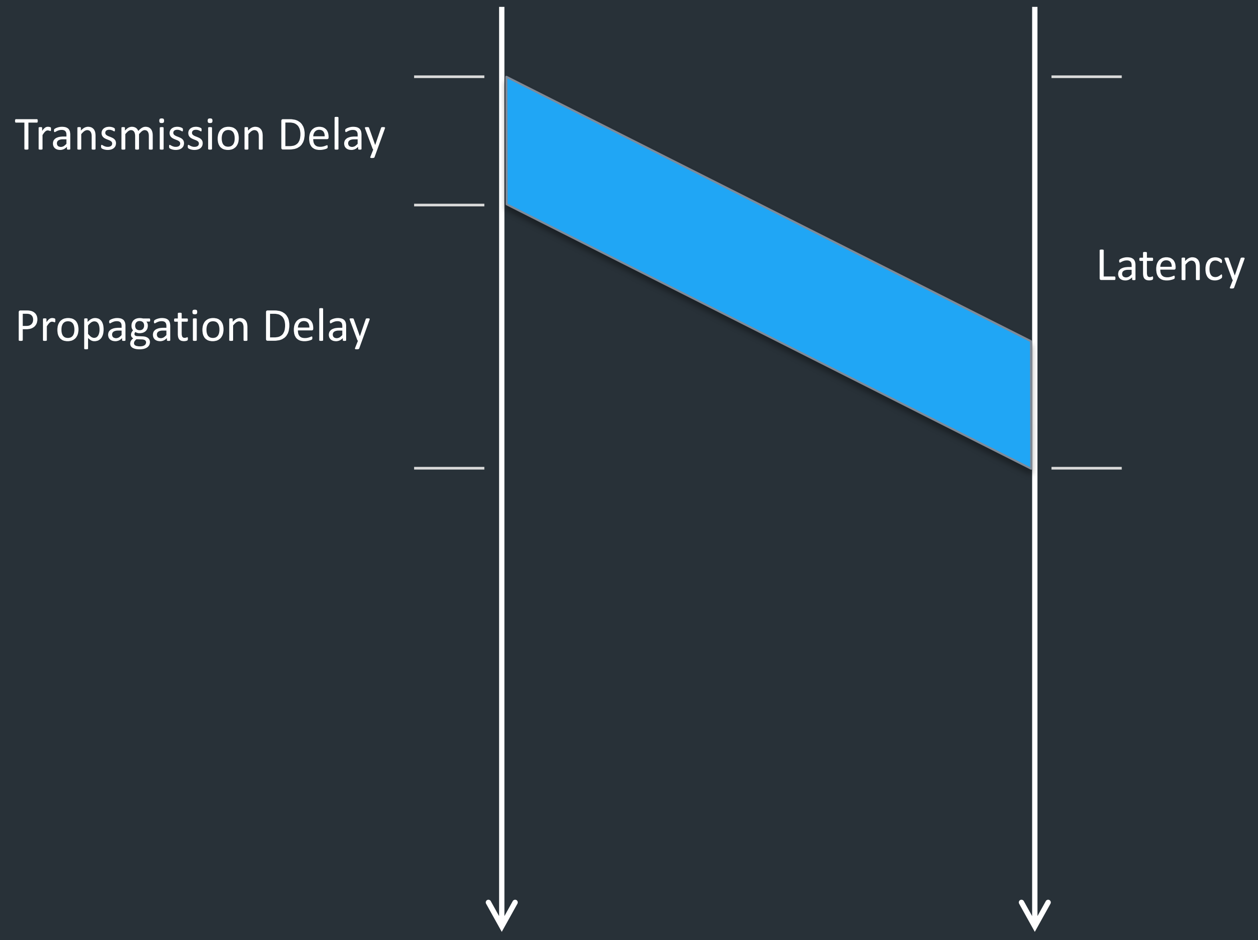
Latency

Sending data takes time!

- Latency: time between sending data and when data arrives (somewhere)

- Multiple components => many definitions, depending on what we're measuring

Sending Frames Across



How to think about latency

How to think about latency

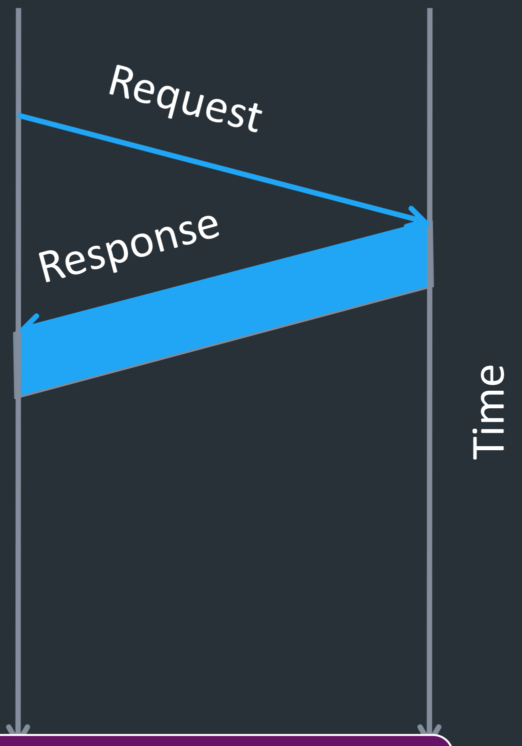
- Processing delay at the node: per message computation
- Queuing delay: time spent waiting in buffers
- Transmission delay: sending out the actual data
  - Size/Bandwidth
- Propagation delay: time for bits to actually go out on the wire
  - Upper bound?
  - Depends on media, ultimate upper bound is speed of light

Ping

Round trip time (RTT): time between request and response

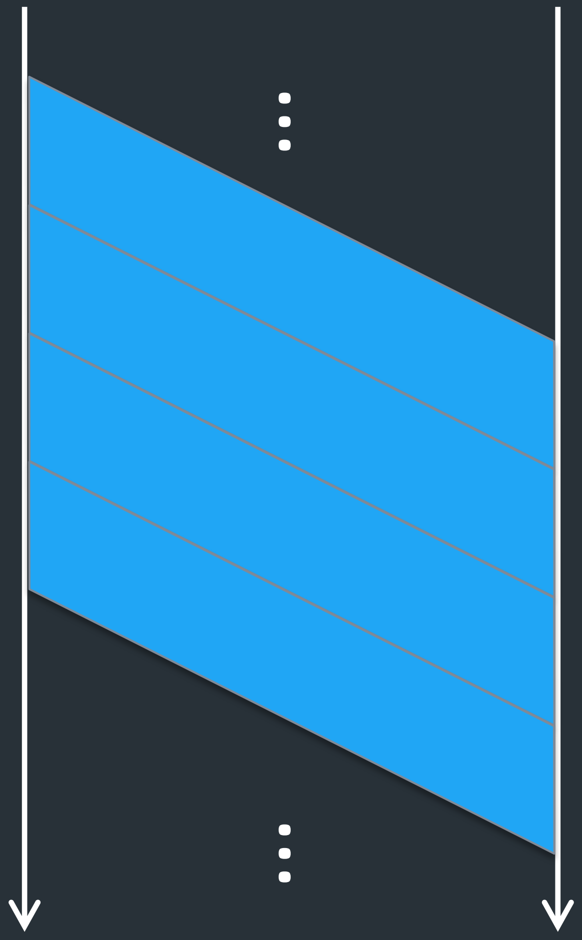
<u>Round trip time (RTT)</u>: time between request and response

When we design protocols, can think about performance based on number of RTTs



=> Not just about the physical layer!

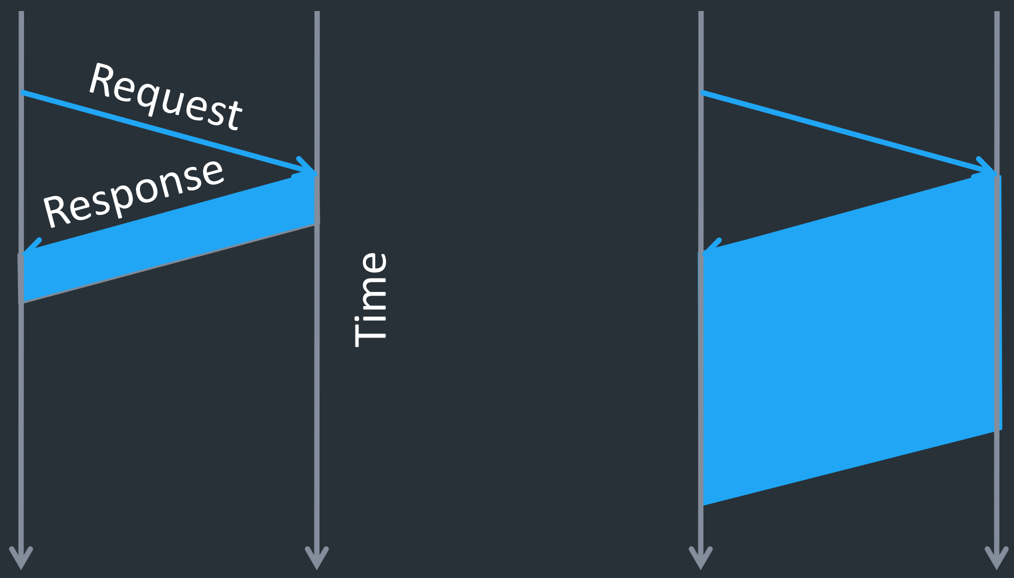
Sending Frames Across



Throughput: bits / s

Which matters most, bandwidth or delay?

- How much data can we send during one RTT?
- *E.g.*, send request, receive file



For small transfers, latency more important,
for bulk, throughput more important

Performance Metrics

- <u>Throughput</u>: Number of bits received/unit of time
  - *e.g.* 100 Mbps
- <u>Goodput</u>: *Useful* bits received per unit of time
- <u>Latency</u>: How long for message to cross network
- <u>Jitter</u>: Variation in latency

Error Detection and Correction

Error Detection

- Basic idea: use a checksum
  - Compute small check value, like a hash of packet

- Good checksum algorithms
  - Want several properties, *e.g.*, detect any single-bit error
  - Details later

Error Detection

- Idea: have some codes be *invalid*
  - Must add bits to catch errors in packet
- Sometimes can also *correct* errors
  - If enough redundancy
  - Might have to retransmit
- Used in multiple layers

Simplest Schemes

- Example: send each bit 3 times
  - Valid codes: 000 111
  - Invalid codes : 001 010 011 100 101 110
  - Corrections : 0 0 1 0 1 1

Parity

Add a *parity bit* to the end of a word

- Example with 2 bits:
  - Valid: 000 011 101 110
  - Invalid: 001 010 010 111
  - Can we correct?
- Can detect odd number of bit errors
  - No correction

In general

Hamming distance: number of bits that are different between two codes

- E.g.: HD (00001010, 01000110) = 3

- If min HD between valid codewords is *d*:
  - Can detect *d*-1 bit error
  - Can correct $\lfloor (d-1)/2 \rfloor$ bit errors

- What is *d* for parity and 3-voting?

Checksums

Compute a “hash” over the message, send with message

Components of Latency

- Processing
  - Per message, small, limits throughput
  - *e.g.* $\frac{100Mb}{s} \times \frac{pkt}{1500B} \times \frac{B}{8b} \approx 8{,}333\,pkt/s$ or 120µs/pkt
- Queue
  - Highly variable, offered load vs outgoing b/w
- Transmission
  - Size/Bandwidth
- Propagation
  - Distance/Speed of Light

Reliable Delivery

- Several sources of errors in transmission
- Error detection can discard bad frames
- Problem: if bad packets are lost, how can we ensure reliable delivery?
  – Exactly-once semantics = at least once + at most once

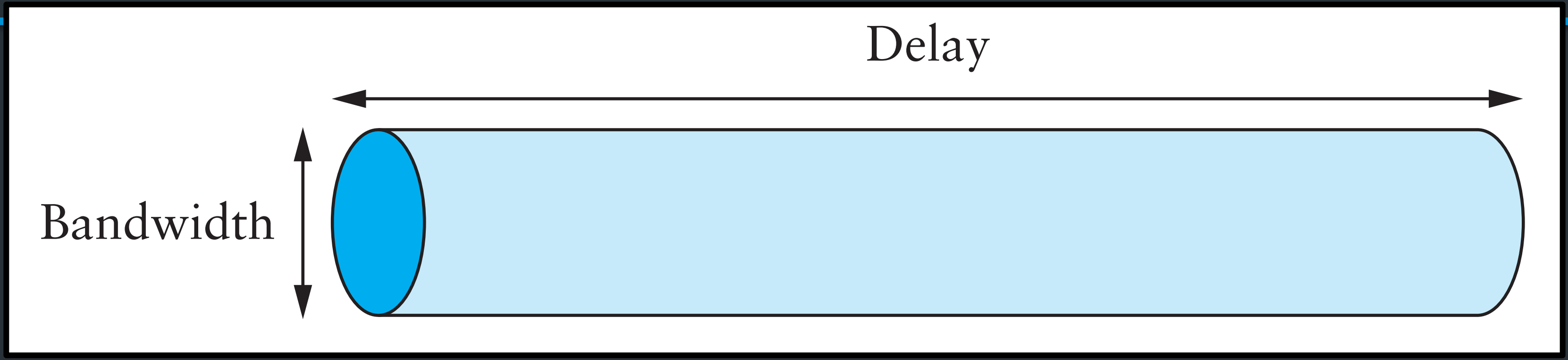
On reliable delivery

- Many link layer protocols don't account for reliable delivery!
  - Eg. Wifi does, Ethernet does not

- Usually, reliable delivery guaranteed by other protocol layers if needed, such as TCP

- Why might we NOT want reliable delivery at the link layer?

Maximizing Throughput



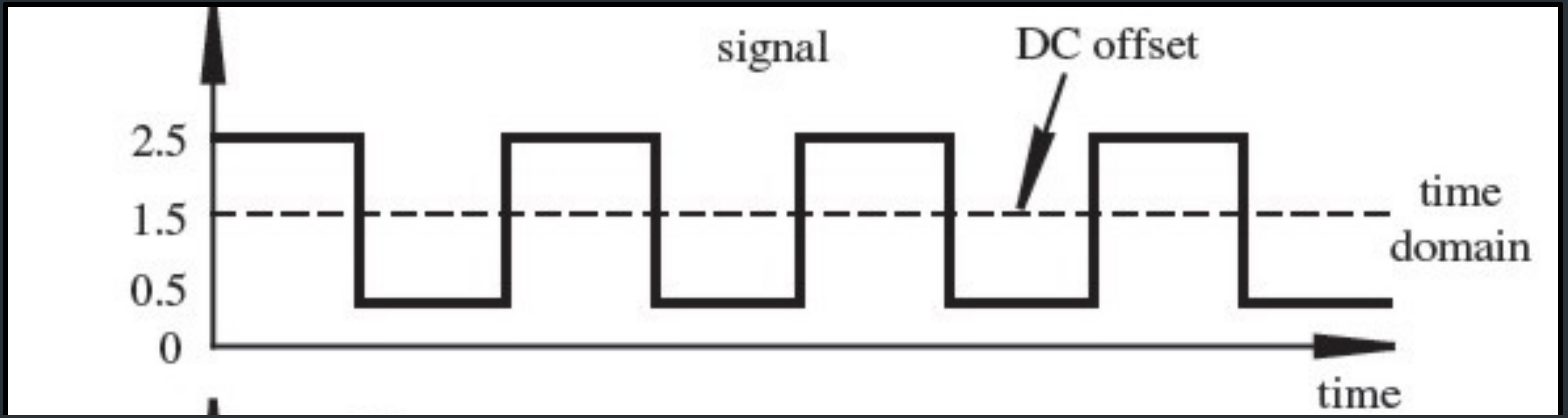
- Can view network as a pipe
  - For full utilization want bytes in flight ≥ bandwidth × delay
  - But don’t want to overload the network (future lectures)
- What if protocol doesn’t involve bulk transfer?
  - Get throughput through concurrency – service multiple clients simultaneously

Summary: Reliable delivery

- Want exactly once
  - At least once: acks + timeouts + retransmissions
  - At most once: sequence numbers
- Want efficiency
  - Sliding window

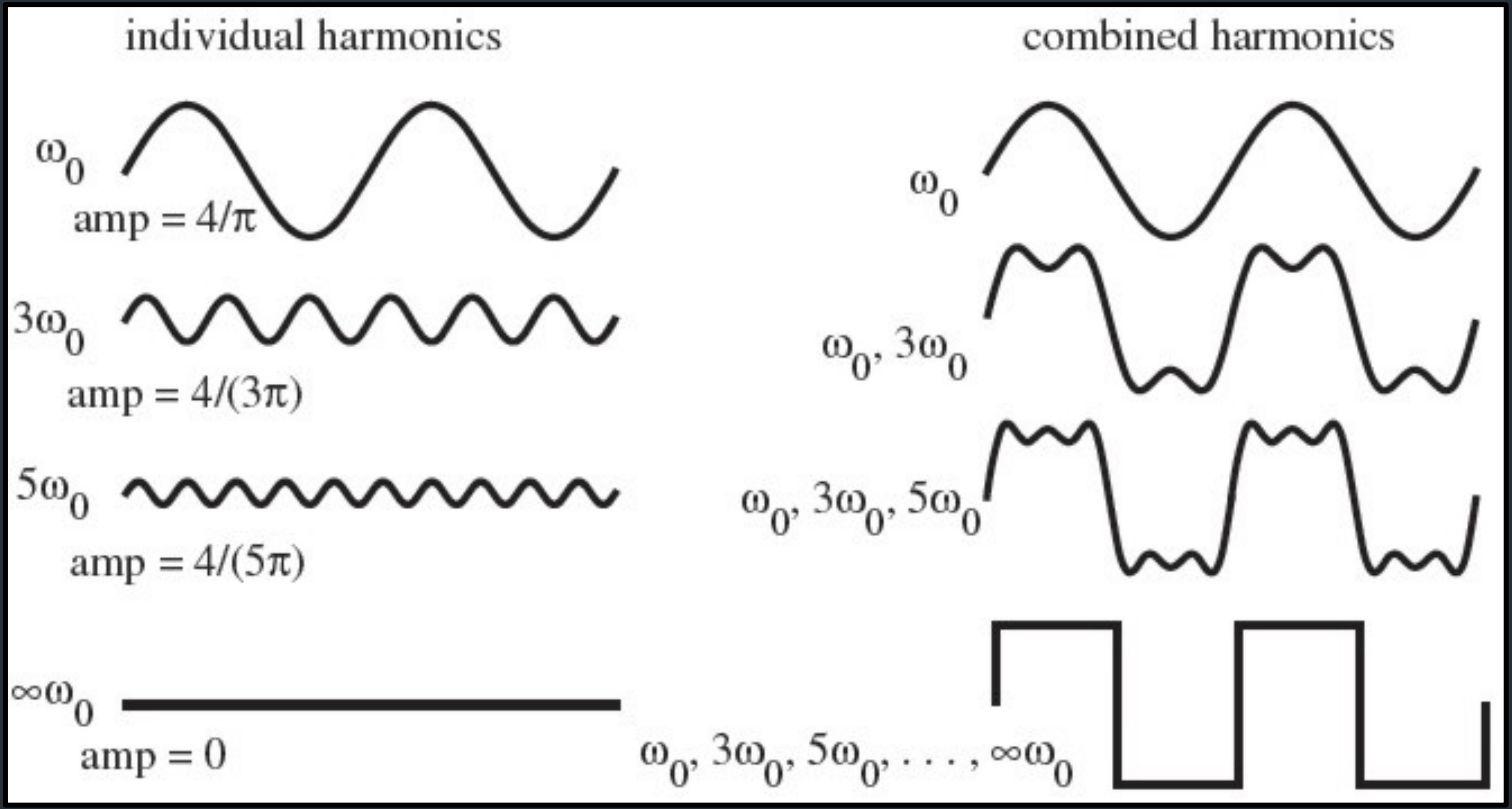
Extra content

Components of a Square Wave



Graphs from Dr. David Alciatore, Colorado State University

Approximation of a Square Wave



Graphs from Dr. David Alciatore, Colorado State University

Can we do better?

- Suppose channel passes 1KHz to 2KHz
  - 1 bit per sample: alternate between 1KHz and 2KHz
  - 2 bits per sample: send one of 1, 1.33, 1.66, or 2KHz
  - Or send at different amplitudes: A/4, A/2, 3A/4, A
  - n bits: choose among 2^n frequencies!

What is the capacity if you can distinguish M levels?

Hartley's Law

$$C = 2B \log_2(M) \text{ bits/s}$$

Great. By increasing M, we can have as large a capacity as we want!

Or can we?

The channel is noisy!

Putting it all together

- Noise limits M!

$$2B \log_2(M) \leq B \log_2(1 + S/N)$$
$$M \leq \sqrt{1+S/N}$$

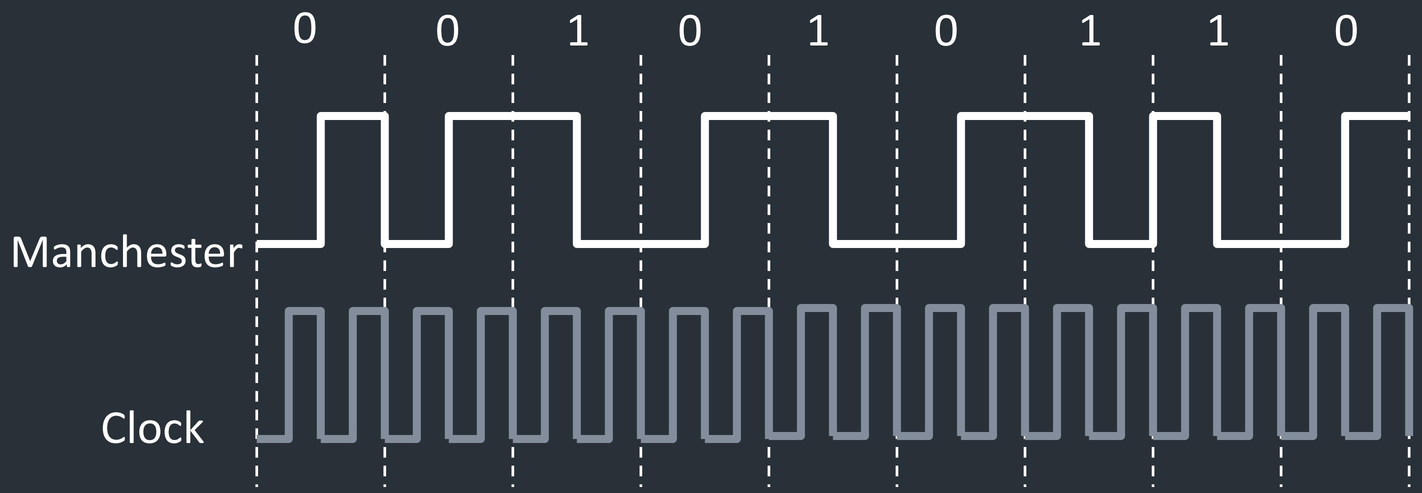
Example: Telephone Line has 3KHz BW, 30dB SNR

- S/N = 10ˆ(30 dB/10) = 1000
- C = 3KHz $\log_2$(1 + 1000) ≈ 30Kbps
- M < sqrt(1001) ≈ 31 levels

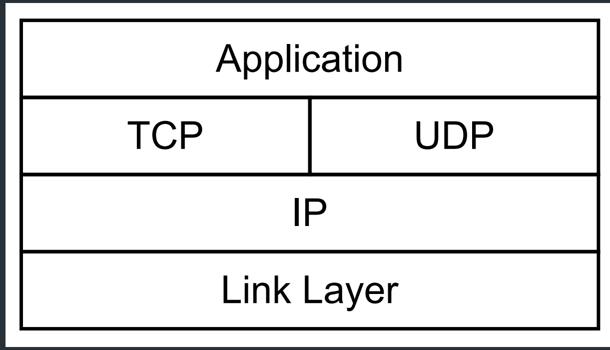
Signal-to-noise ratio (SNR)
is typically measured in Decibels (dB)
dB = $10\log_{10}(S/N)$

Manchester Encoding

- Map 0 → 01; 1 → 10
  - Transmission rate now 1 bit per two clock cycles
- Solves clock recovery & baseline wander
- … but halves transmission rate!



Layering

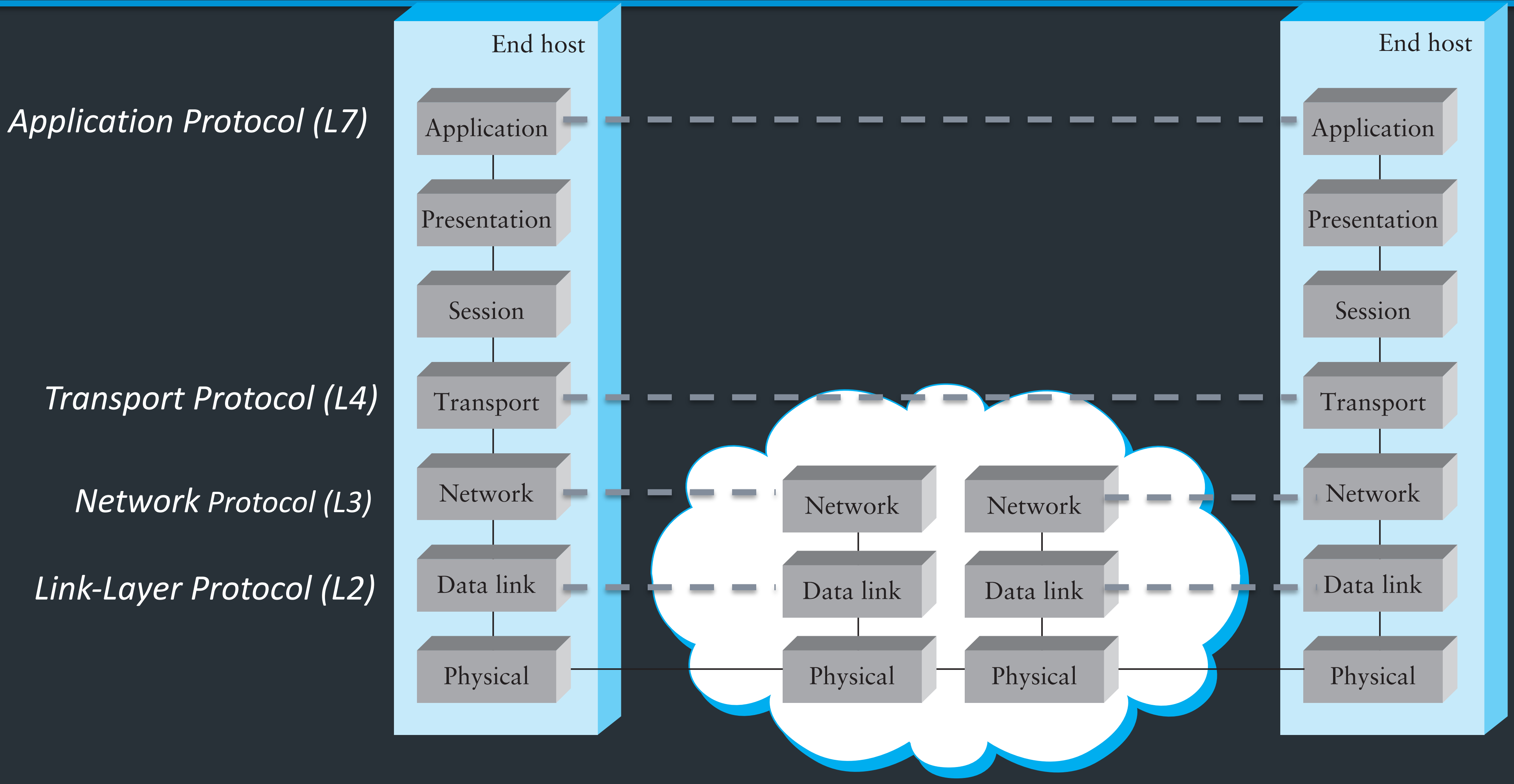


Abstraction to the rescue!

- Break problem into separate parts, solve part independently
- Abstract data from the layer above inside data from the layer below

Encapsulate data from "higher layer" inside "lower layer"
=> Lower layer can handle data without caring what's above it!

The big complex picture



"OSI reference model" or "7-layer model"

Applications (Layer 7)

Application	
TCP	UDP
IP	
Link Layer	

The applicatons/programs/etc you use every day

Examples:

- HTTP/HTTPS: Web traffic (browser, etc)
- SSH: secure shell
- FTP: file transfer
- DNS (more on this later)
- …

When you're building programs, you usually work here