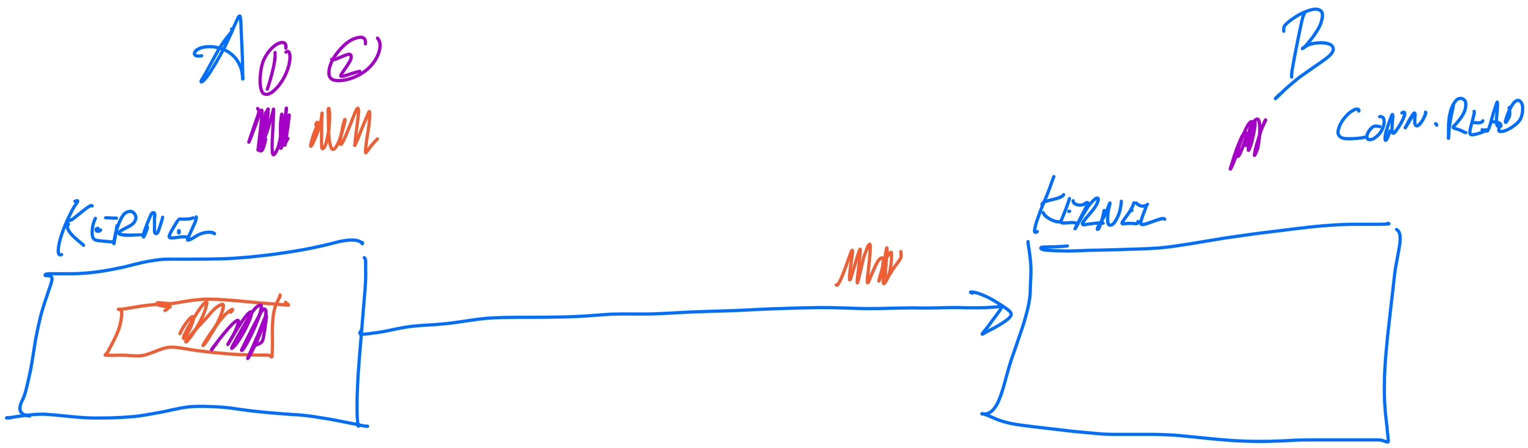
CSCI 1680
Physical Layer, Link Layer I

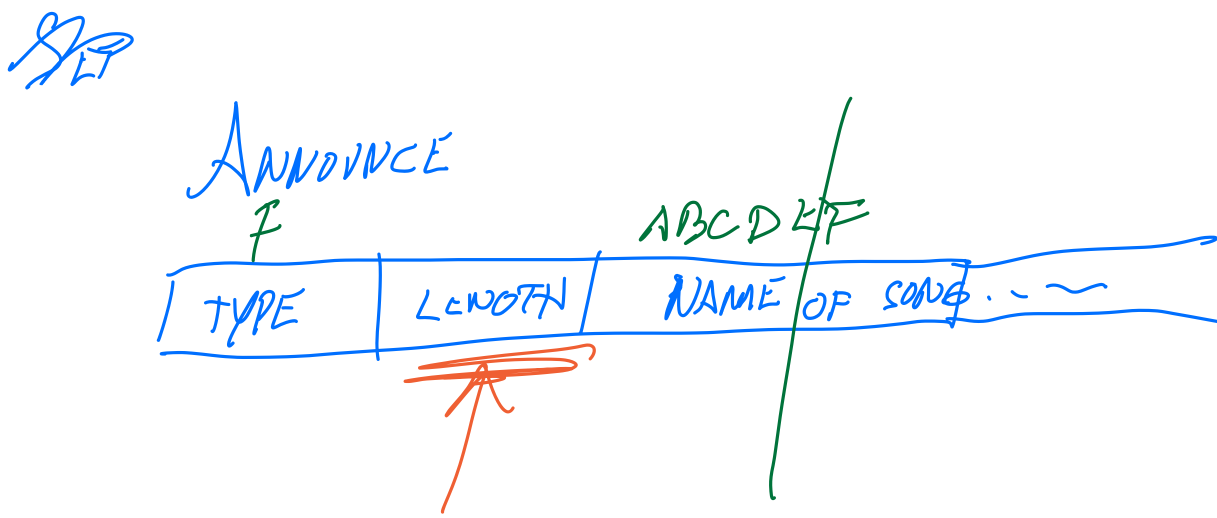
Nick DeMarinis

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



In TCP, conn.Read returns when ANY AMOUNT of data is available
=> MAY NOT be the amount you want

Solution: keep reading until you get the amount of data that you want io.ReadFull



Sending Data w/ TCP


```
for {
  bytes_read = read(...)
}
```

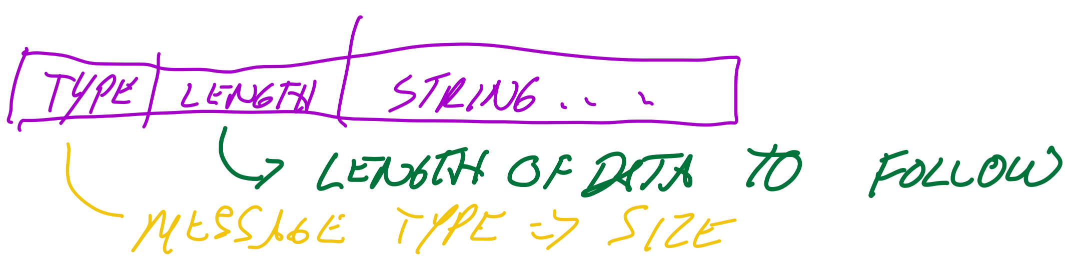
When you read on a TCP socket, you might not get back the amount of data you expect => need to check and act accordingly
Idea: call read in a loop until you get back how much you wanted
In Go: io.ReadFull

TCP is designed to provide a STREAM of ordered data => it doesn't care about the separation of individual messages

What happens if you don't know the size of the message?

=> Protocol needs to be set up so that you can always figure out how much data to read next.

E.G.



Today

- Two more things on sockets

- Physical/Link layer: how to connect two things

  => Inherent properties of *real* networks

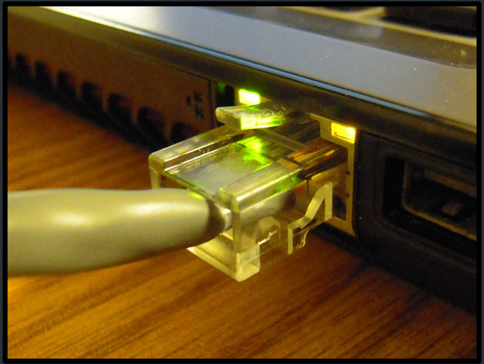
Layers, Services, Protocols

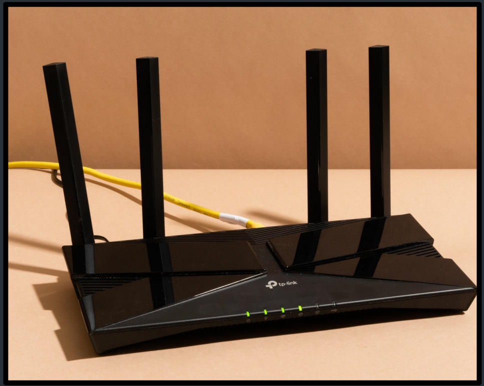
Application	Service: user-facing application. Application-defined messages
Transport	Service: multiplexing applications Reliable byte stream to other node (TCP), Unreliable datagram (UDP)
Network	Service: move packets to any other node in the network IP: Unreliable, best-effort service model
L2 Link	Service: move frames to other node across link. May add reliability, medium access control
L1 Physical	Service: move bits to other node across link

Physical Layer (Layer 1)

Specifies three things:

- Physical medium: WIFI, ETHERNET, CELLULAR...
- Signaling/modulation: HOW DO I SEND 0 OR 1
- Encoding: → HOW DO YOU GET USEFUL INFO

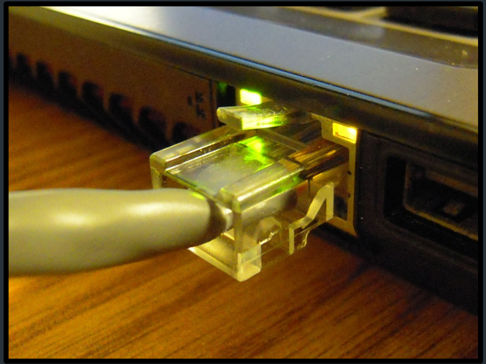


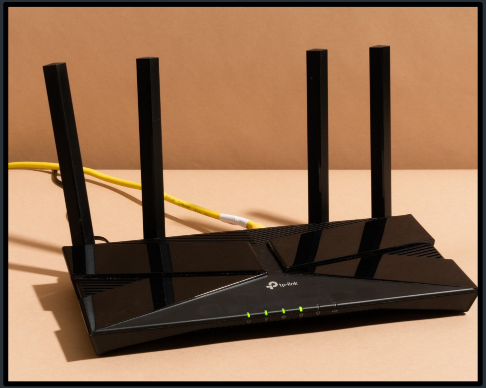



Physical Layer (Layer 1)

Specifies three things:

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





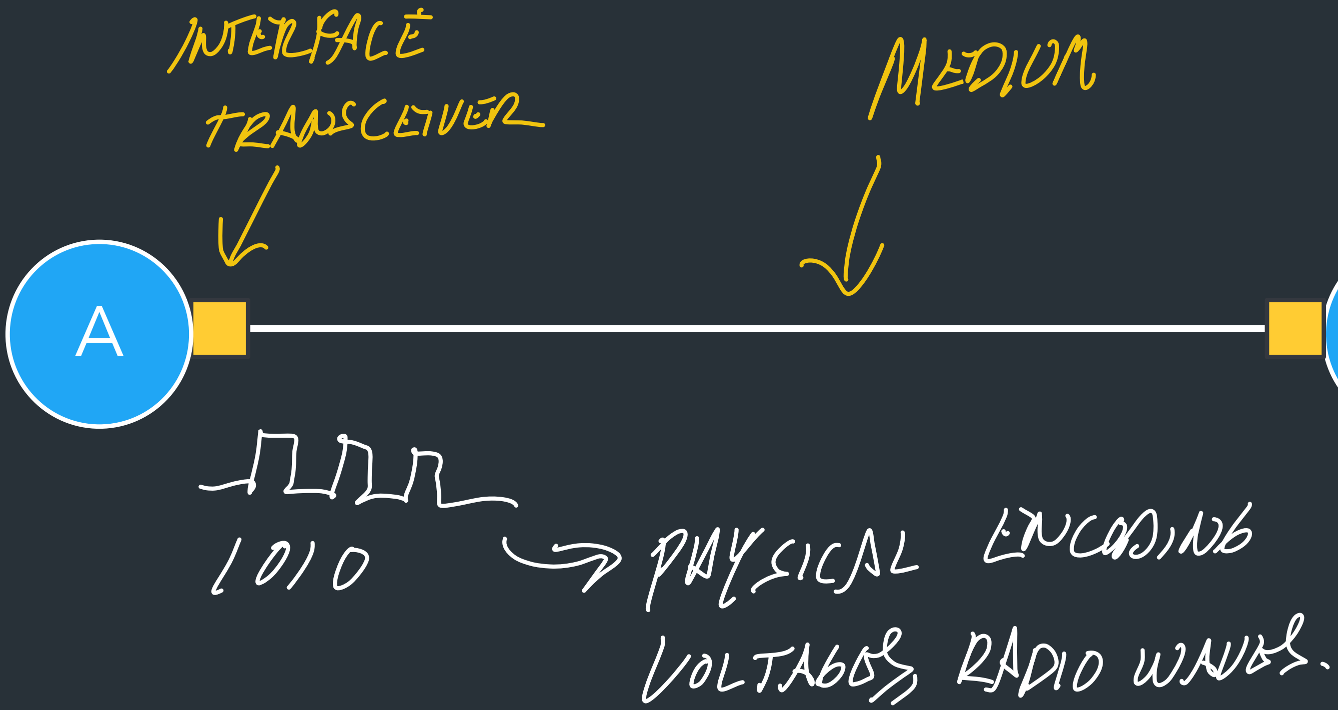
Why should we care?

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 some kind of medium
Sender encodes message in some format, sends it "over the wire"
Receiver decodes message (knowing the format) => recover the message

Why is this hard?

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

=> Every medium has its own characteristics, and problems

We don't need to know the details.

However, there are some key takeaways to help understand the challenges and implications

Key points

1. All media have fixed bandwidth => fixed “space” to transmit information

2. Sending data takes time! => latency

3. All media have (some) errors => how to deal with them?

Bandwidth

Bandwidth: set of frequencies that a channel (medium) can propagate wall

(Hz)

PHONE/DIALUP: 8kHz

- Creates a fixed "space" in which data can be transmitted

=>Wires: defined by physical properties

⇒ Wireless: frequency ranges are regulated

Bandwidth gives an upper bound on throughput => amount of data we can send per time (bits / second)

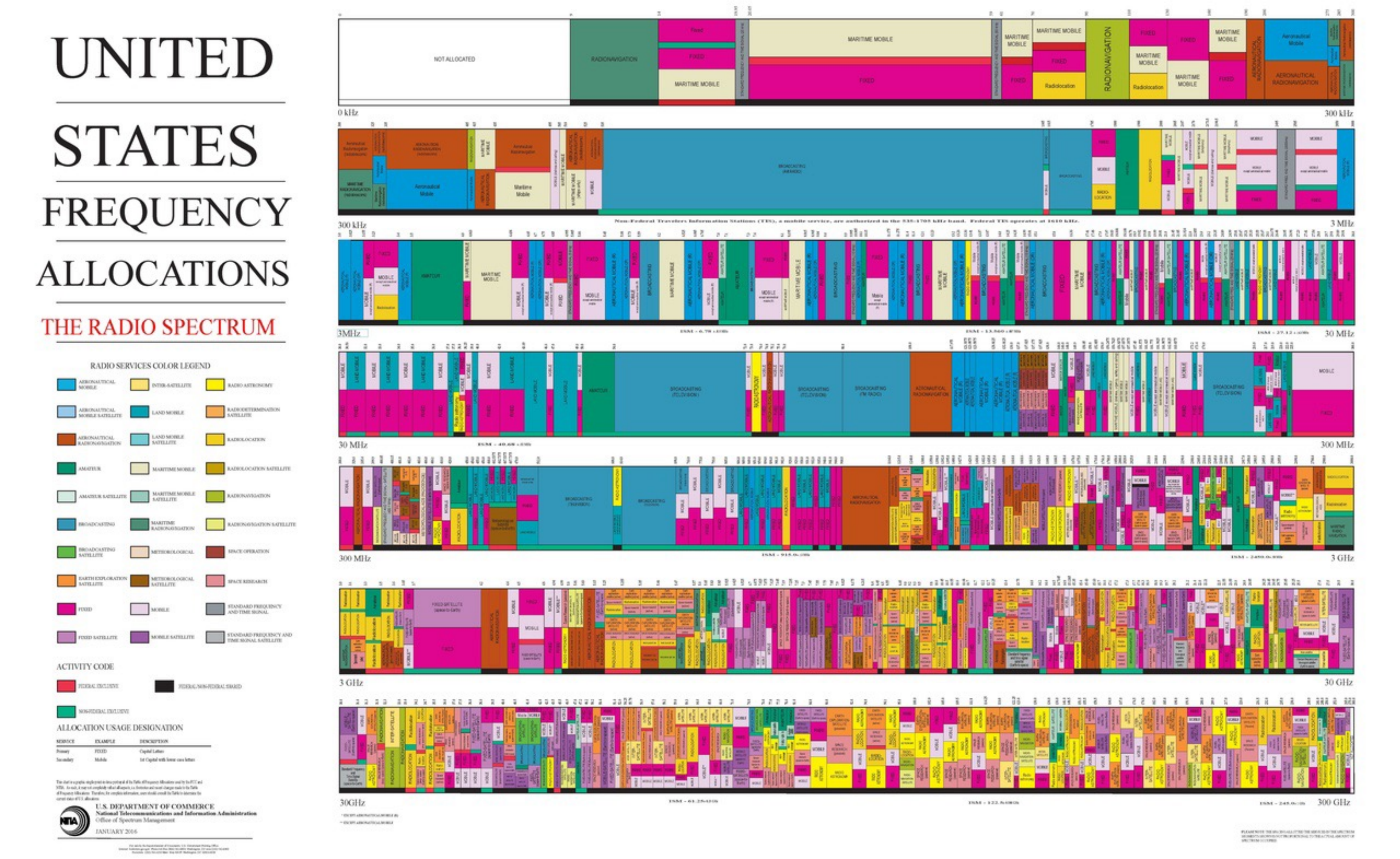
Bandwidth: frequencies that a channel propagates well
(Most signals made up of different frequencies)

- Creates a fixed "space" in which data can be transmitted

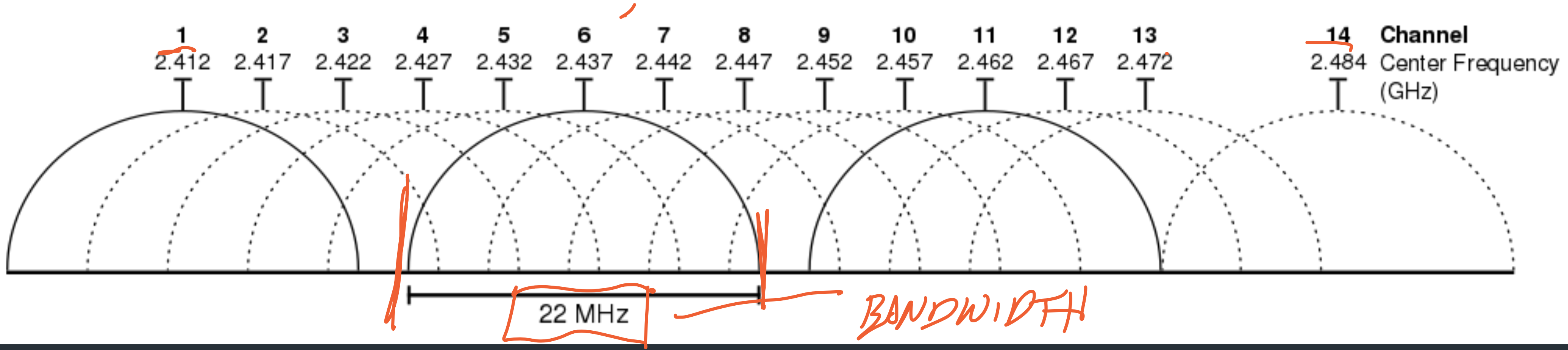
  =>Wires: defined by physical properties

  ⇒ Wireless: frequency ranges are regulated

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

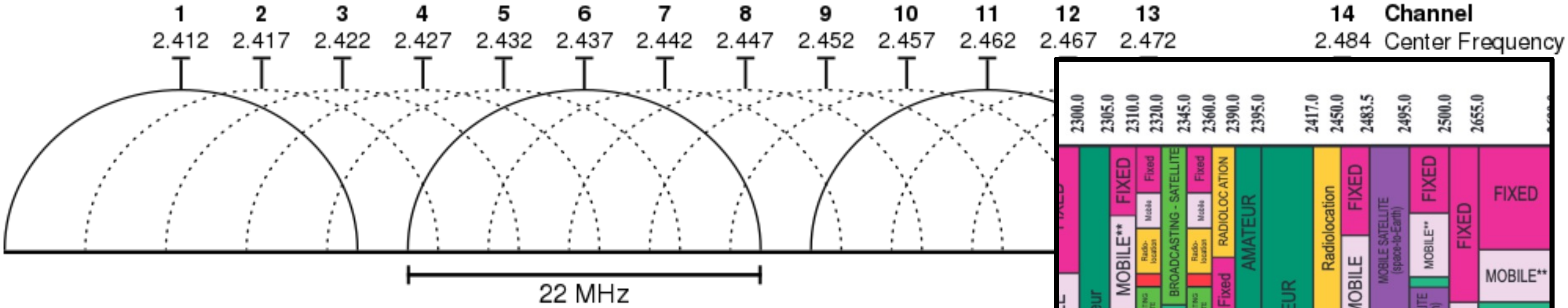

UNITED
STATES
FREQUENCY
ALLOCATIONS
THE RADIO SPECTRUM
RADIO SERVICES COLOR LEGEND
AERONAUTICAL MOBILE
INTER-SATELLITE
RADIO ASTRONOMY
AERONAUTICAL MOBILE SATELLITE
LAND MOBILE
RADIODETERMINATION SATELLITE
AERONAUTICAL RADIONAVIGATION
LAND MOBILE SATELLITE
RADIOLOCATION
AMATEUR
MARITIME MOBILE
RADIOLOCATION SATELLITE
AMATEUR SATELLITE
MARITIME MOBILE SATELLITE
RADIONAVIGATION
BROADCASTING
MARITIME RADIONAVIGATION
RADIONAVIGATION SATELLITE
BROADCASTING SATELLITE
METEOROLOGICAL
SPACE OPERATION
EARTH EXPLORATION SATELLITE
METEOROLOGICAL SATELLITE
SPACE RESEARCH
FIXED
MOBILE
STANDARD FREQUENCY AND TIME SIGNAL
FIXED SATELLITE
MOBILE SATELLITE
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
U.S. DEPARTMENT OF COMMERCE
National Telecommunications and Information Administration
Office of Spectrum Management
JANUARY 2016
0 kHz
300 kHz
3 MHz
3MHz
30 MHz
300 MHz
3 GHz
30 GHz
30GHz
300 GHz
NOT ALLOCATED
RADIONAVIGATION
MARITIME MOBILE
FIXED
AERONAUTICAL RADIONAVIGATION
Aeronautical Mobile
Radiolocation
BROADCASTING (AM RADIO)
AMATEUR
BROADCASTING (TELEVISION)
BROADCASTING (FM RADIO)
MOBILE

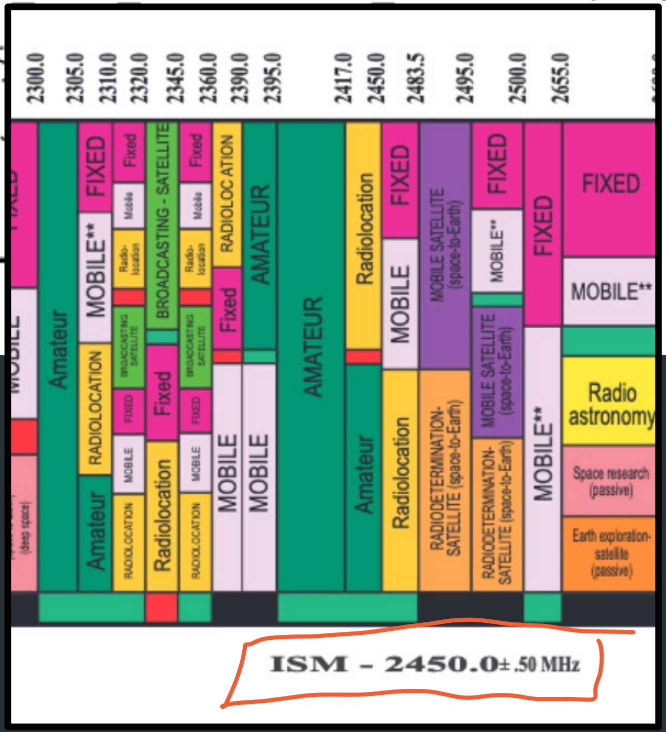
Early IEEE 802.11 (Wifi) channel bandwidth



2.4 GHz : "UNLICENSED BAND"

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]

IEEE 802.11

How to actually send data?

(Within a limited bandwidth)

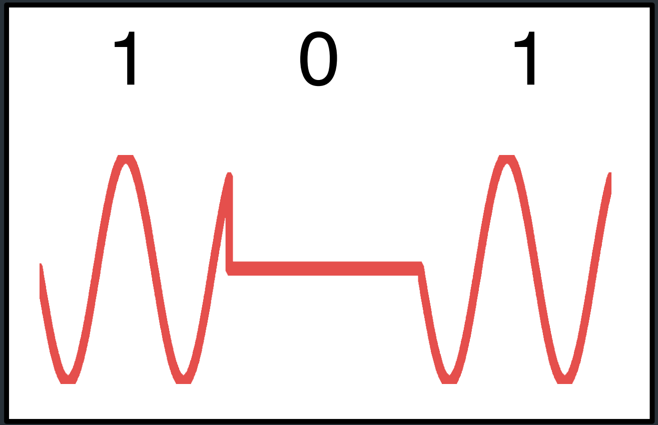
How to actually send stuff?

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

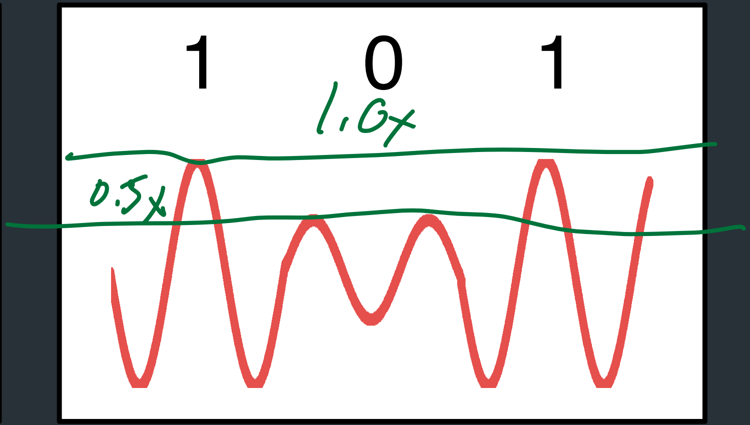
One way: Use Carriers

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

OOK: On-Off Keying



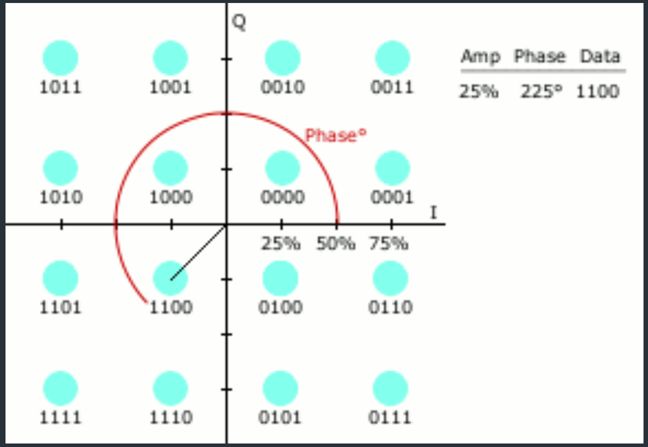
ASK: Amplitude Shift Keying



This can get more complex…

Lots of engineering you can do

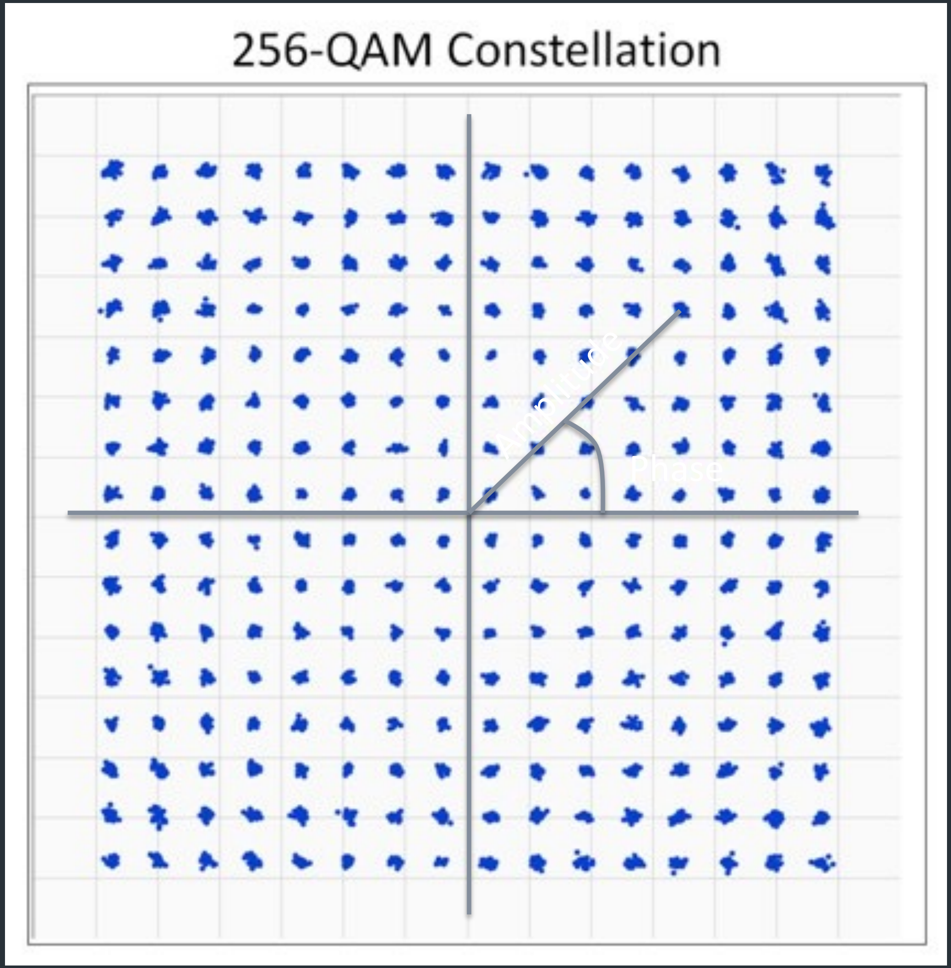
- Multiple carriers/frequencies
- Adjust amplitude, phase
- Clever ways to avoid errors
- …



A good animation on Wikipedia

QAM: Quadrature Amplitude Modulation

Example: Quadrature Amplitude Modulation (QAM)

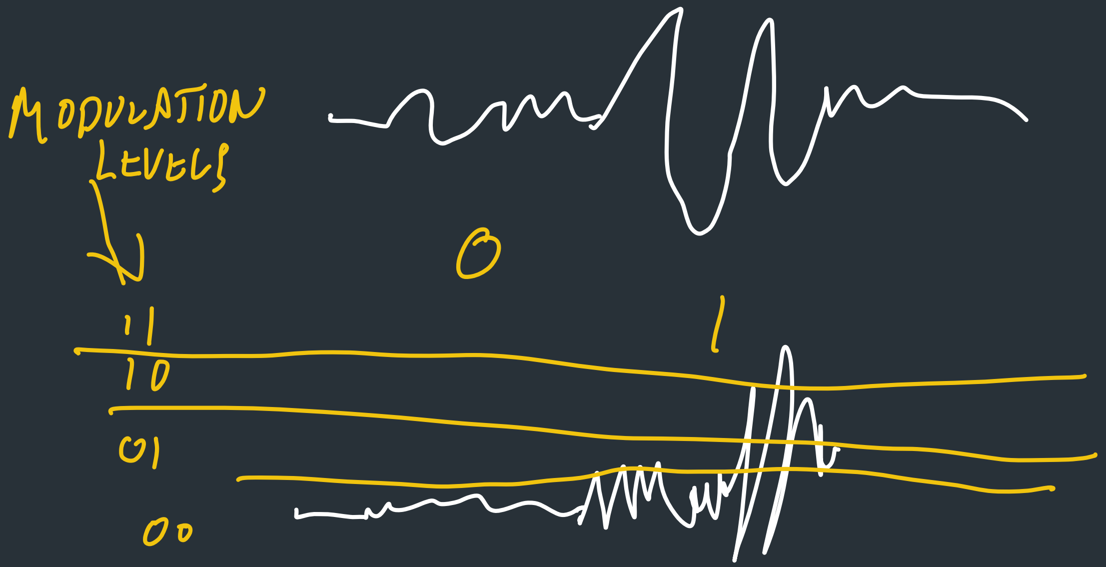


Modulation schemes in action

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

Sounds great, right?

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



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 (throughput) in bits/second
- B : bandwidth in Hz
- S, N: average signal, noise power

The amount of data we can fit in a channel is limited by the bandwidth, and the amount of noise in the medium

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 (throughput) in bits/second
- B : bandwidth in Hz
- S, N: average signal, noise power

Takeaway: fundamental limit on how much data we can fit into a fixed channel, based on noise

=> For any medium, designers create encodings to try and maximize throughput

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

10^{-9}

RANGE

INTERFERENCE

10^{-12}

=> Does this mean wifi is the best?

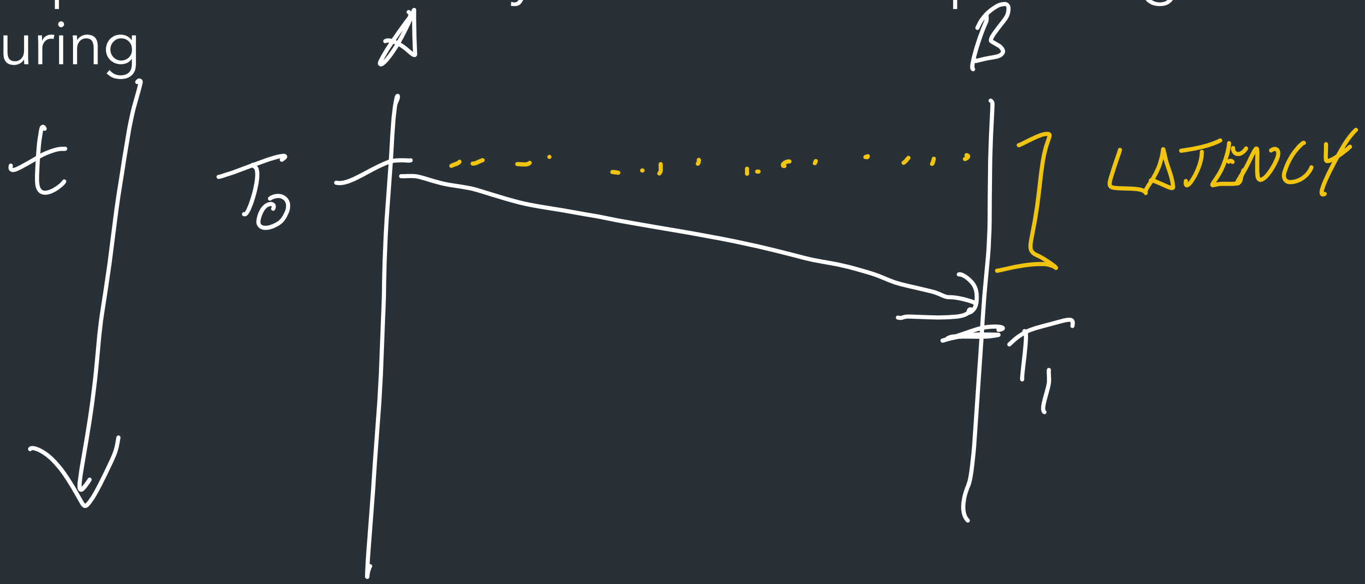
Latency

Sending data takes time!

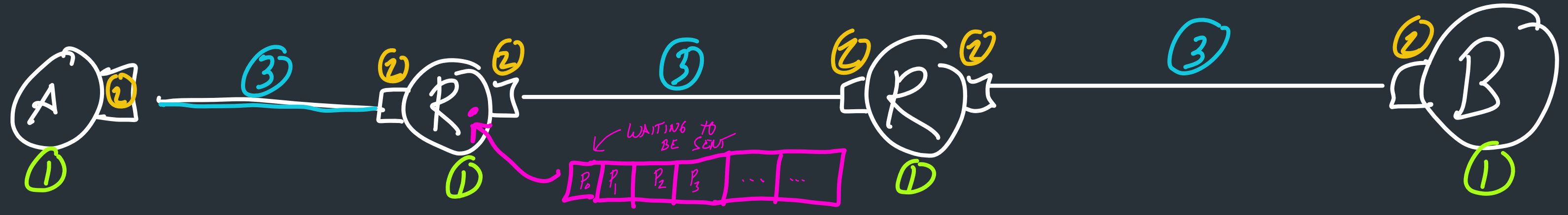
- Latency: time between sending data and when data arrives (somewhere)

Sending data is not instantaneous (for many reasons)

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



How to think about latency



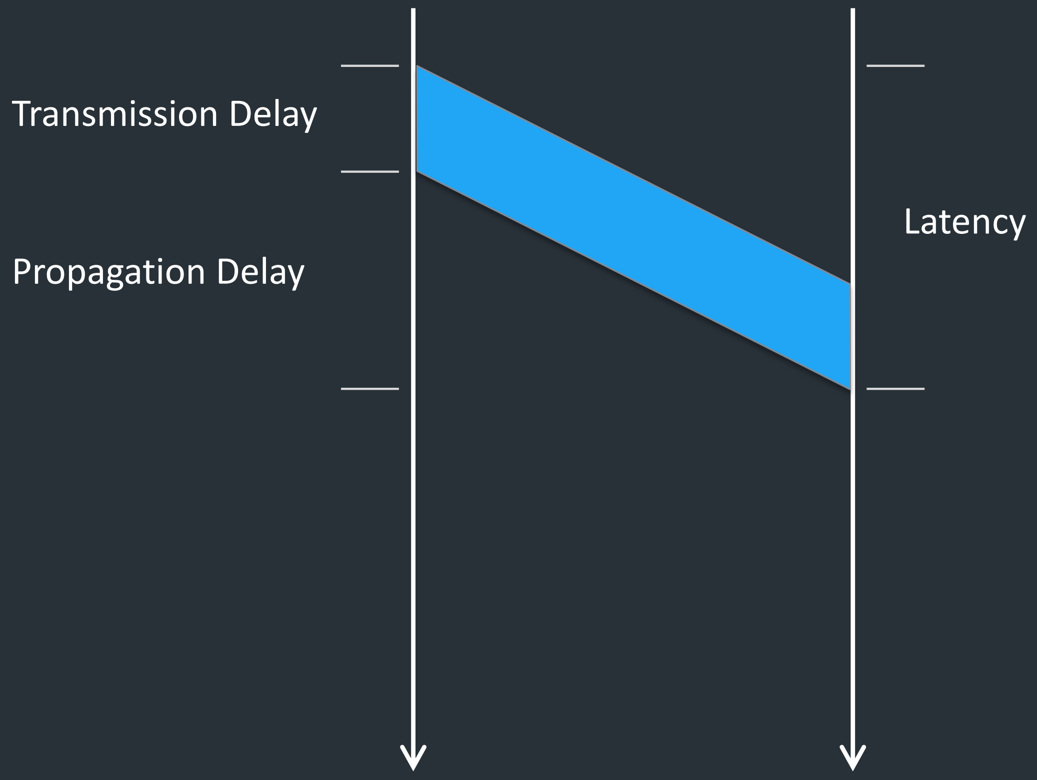
1. Processing delay: Time for data to be sent by a node
   => Time for OS to decide to send data

2. Transmission delay: Time for transceiver to actually send bits on wire

3. Propagation delay: Time for data to propagate across wire => speed of light

4. Queuing delay: time data spends in buffers waiting to be sent (due to congestion, etc.)

How to think about latency

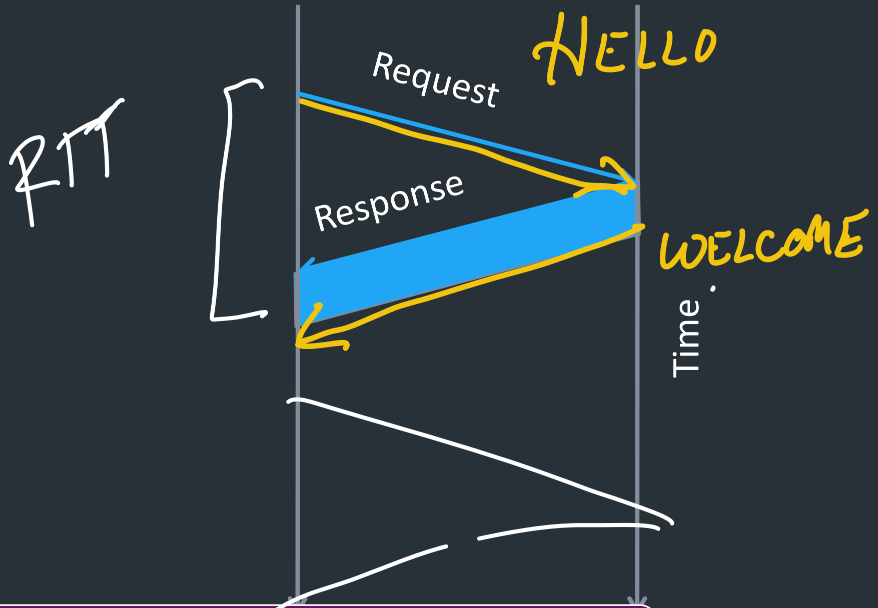


How to think about latency

- 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

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

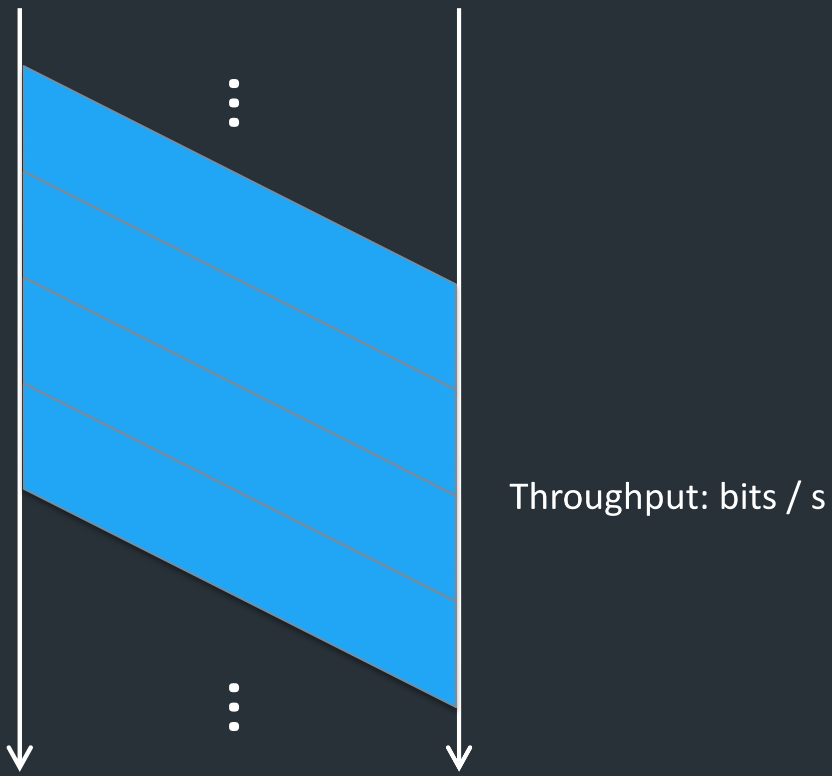
When we design protocols, can think about performance based on number of RTTs



=> Not just about the physical layer!

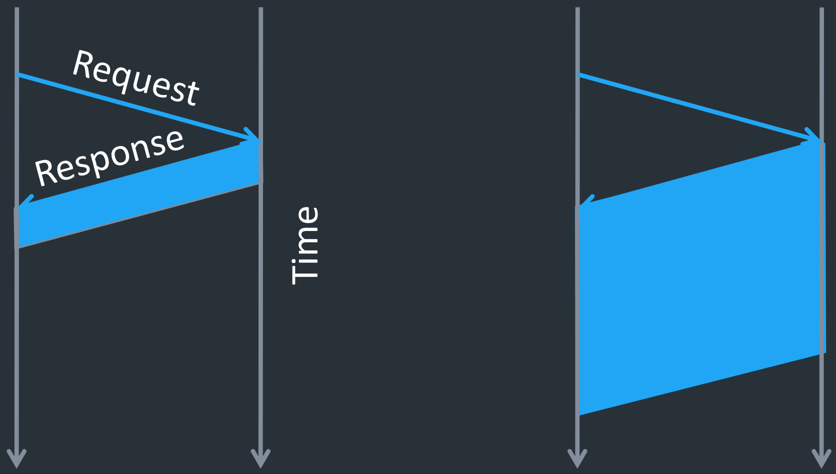
Next few pages are extra material we'll discuss later!

Sending Frames Across



Which matters most, bandwidth or delay?

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



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

Performance Metrics

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

Dealing with errors

Error Detection

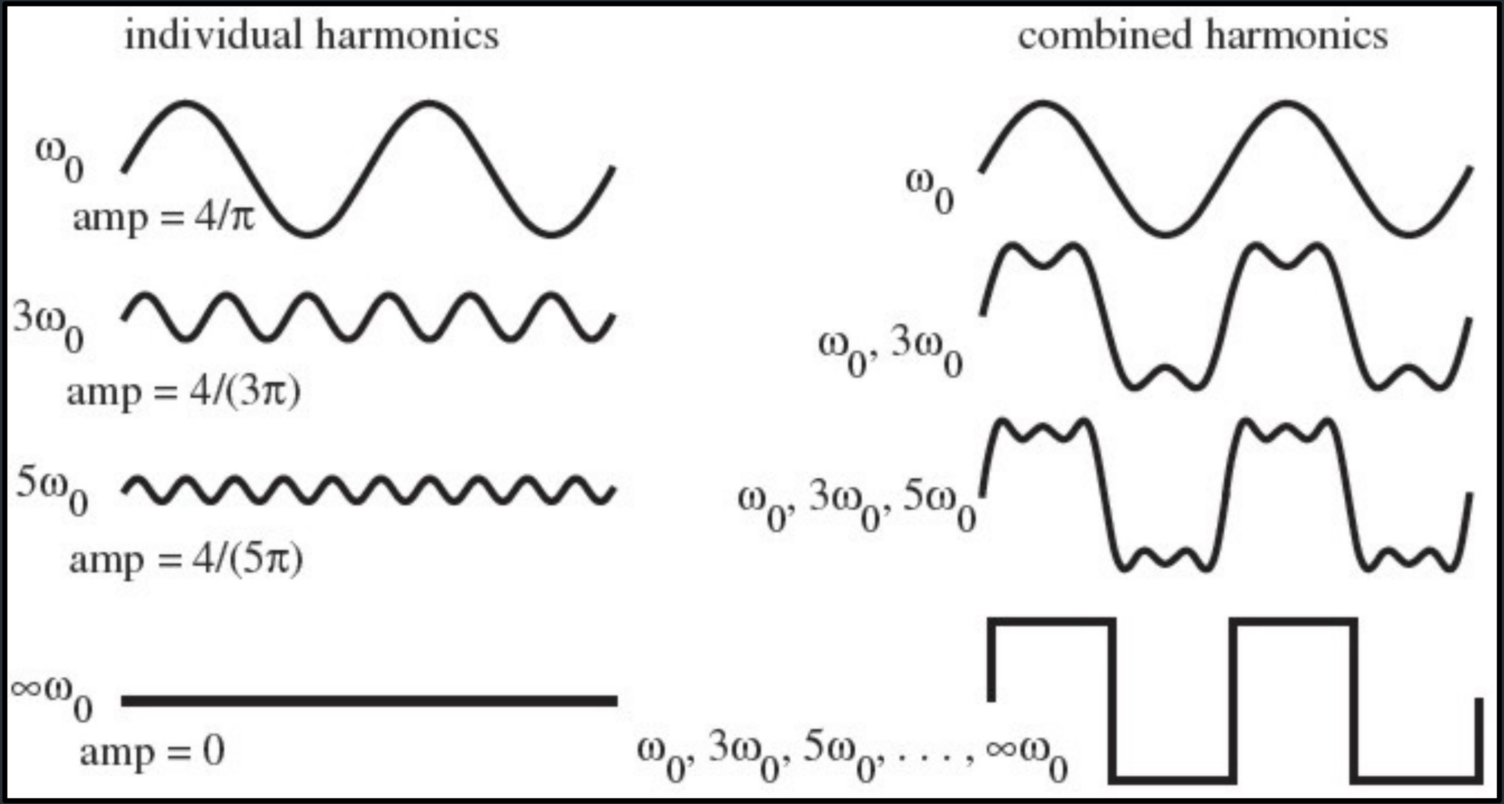
- Basic idea: use a checksum
  - Compute small check value, like a hash of packet

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

=> Not all protocols do this. Why?

Approximation of a Square Wave



Graphs from Dr. David Alciatore, Colorado State University