CSE 461: Midterm Review!

Autumn 2020

Midterm Info

- November 9th, 12:30pm - 1:20pm PST

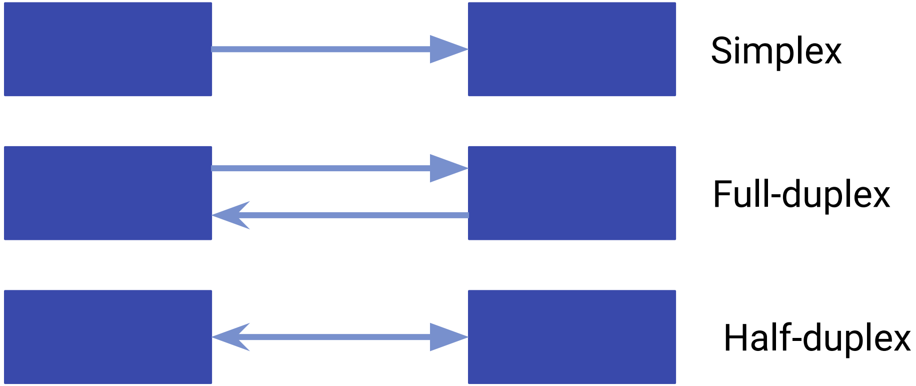
- Reminder: Assignment 3 due tomorrow!

Networks Overview

- Parts of a network
- Types of links
- Key interfaces
- Sockets
- Traceroute
- Protocols and layers
- Encapsulation

Types of Links

- Parts of a network
    - Application, Host, Router, Link
- Types of links
    - Full-duplex, Half-duplex, Simplex



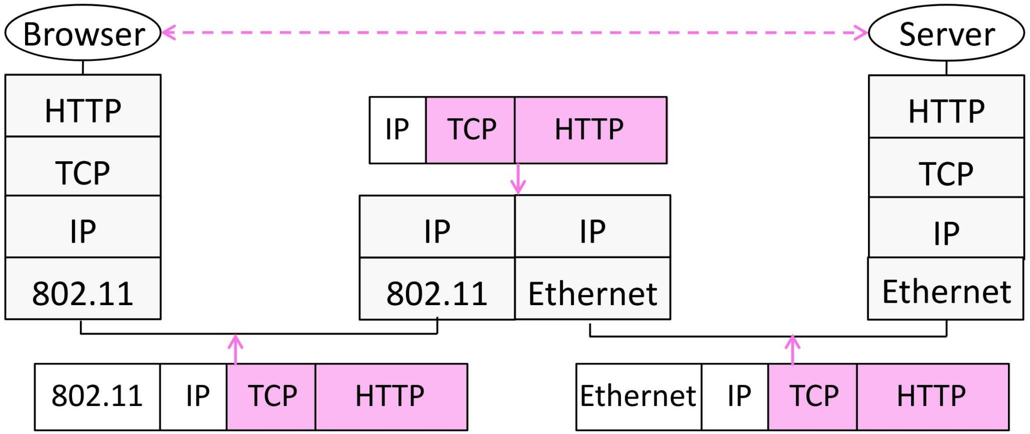
Layering

ADVANTAGES

- Use information hiding to connect different systems
- Information reuse to build new protocols

DISADVANTAGES

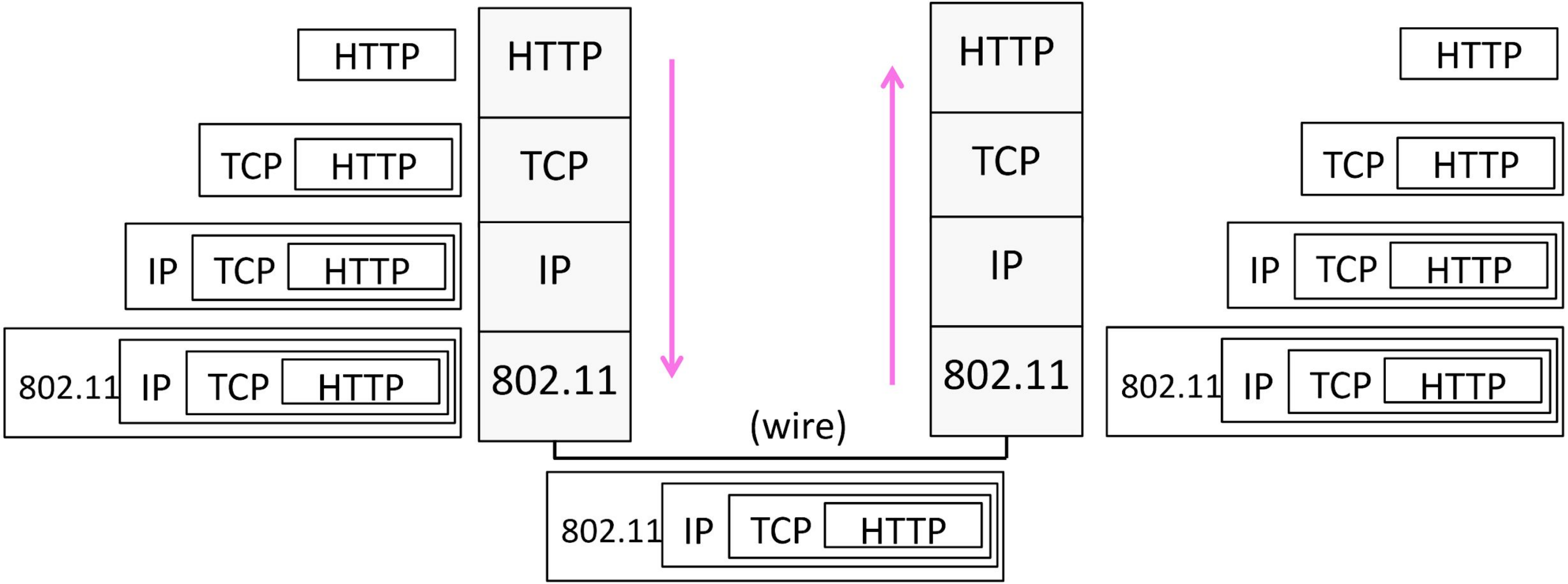
- Adds overhead
- Hides information from lower layers



Protocols and Layers

	Purpose	Protocols	Unit of Data
Application	Programs that use network service	HTTP, DNS	Message
Transport	Provides end-to-end data delivery	TCP, UDP	Segment
Network	Sends packets across multiple networks	IP	Packet
Link	Sends frames across a link	Ethernet, Cable	Frame
Physical	Transmit bits	__	Bit

Encapsulation

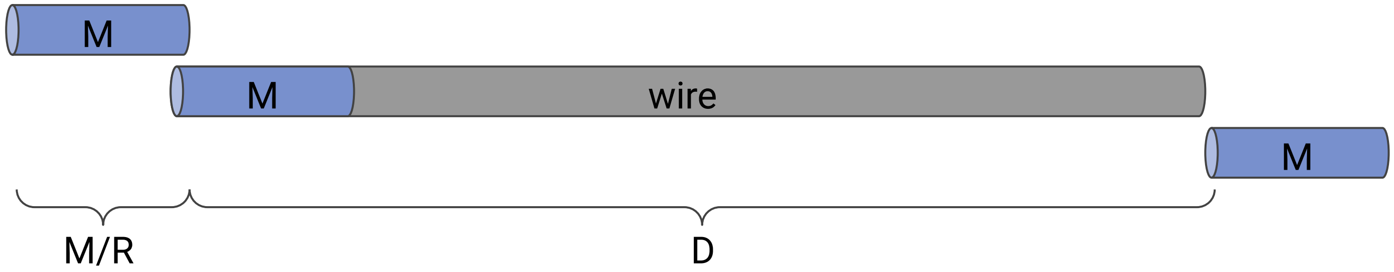


Physical Layer

- Latency
- Types of media
- Signal propagation
- Modulation schemes
- Fundamental limits

Latency

- Latency = Transmission Delay + Propagation Delay
- Transmission Delay = M (bits) / R (bits/sec) = M/R (sec)
- Propagation Delay = Length / Speed of Signals = Length / ⅔c = D (sec)



- Bandwidth-Delay Product = R (bits/sec) x D (sec) = BD (bits)

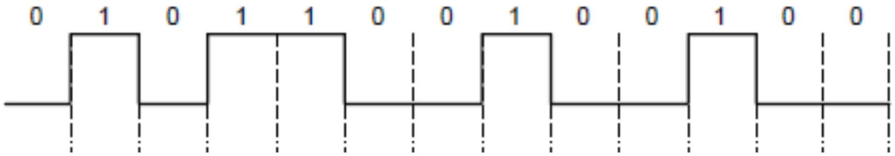
Signal Propagation

- Over wire
  - Signal is delayed
  - Signal is attenuated, especially as distance increases
  - Frequencies above a certain cutoff are highly attenuated, due to limited bandwidth
  - Noise is added to signal
- Over fiber
  - Signal transmitted on carrier frequency
- Over wireless
  - Signal attenuates $1/D^2$
  - Signals on same frequency interfere at receiver
  - Multipath interference at receiver

Modulation Schemes

- Baseband modulation allows signal to be sent directly on wire

NRZ signal of bits

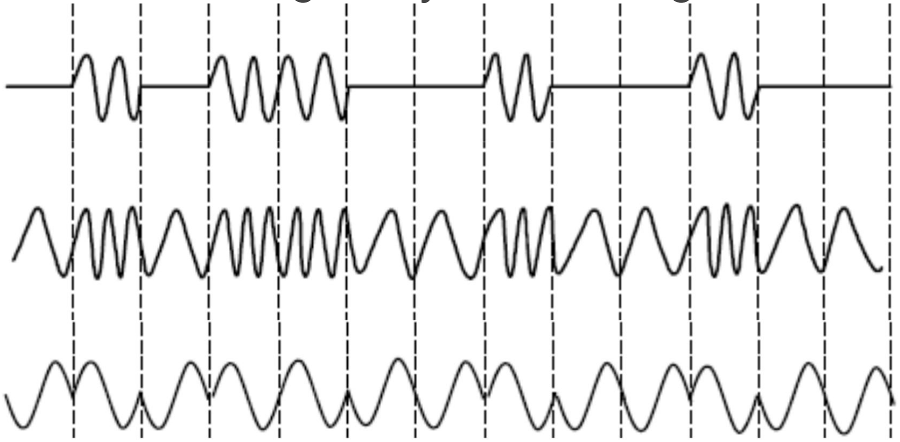


- Passband modulation carries a signal by modulating a carrier

Amplitude shift keying

Frequency shift keying

Phase shift keying

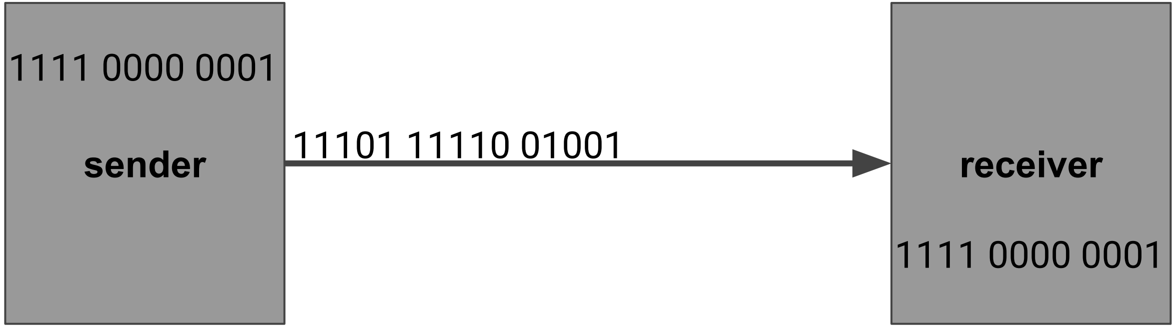


We could also use more than 2 signal levels by using more bits per signal.
2^{bits} = voltage levels

Clock Recovery

100000000000000000000000000000000000000000000000001

- Manchester coding: map every 4 data bits to 5 data bits, such that there are no more than 3 zeros in a row
- Scrambling



Shannon Capacity

What is the maximum information carrying rate of a channel?

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

$$C = B \log_2(1 + S/BN) \text{ bits/sec}$$

How would you increase C if SNR is really poor? $C \rightarrow S/N$

How would you increase C if SNR is really good?

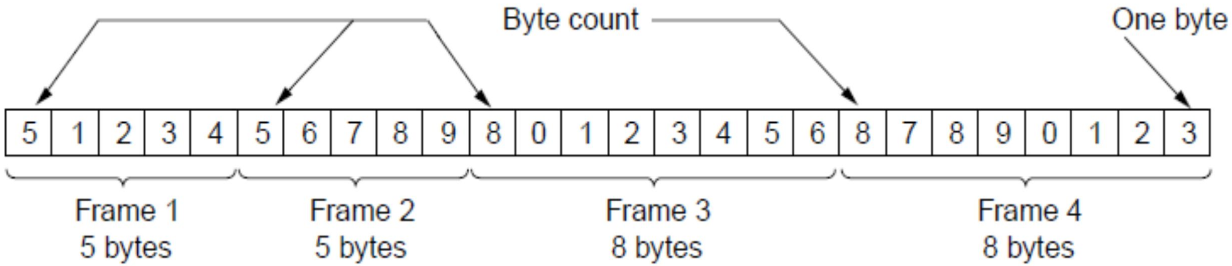
Increasing Bandwidth is MUCH more effective than increasing Signal or decreasing Noise.

Link Layer

- Framing
- Error detection and correction
- Multiplexing
- Multiple access control
- Switching

Framing Methods

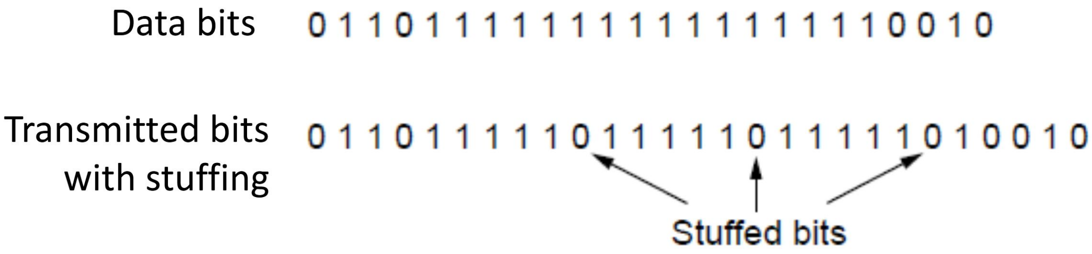
- How do we know where a bit sequence (frame) begins and ends?
  - Byte count
  - Byte stuffing
  - Bit stuffing
- Byte Count
  - Problem: How to find start of frame if there is an error?



Framing Methods

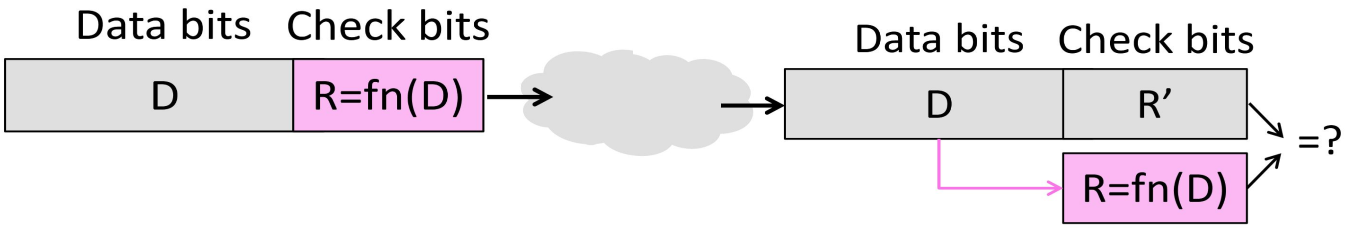
- Byte Stuffing
    - Replace ESC in data with ESC ESC, and replace FLAG in data with ESC FLAG

- Bit Stuffing
    - Less overhead than byte stuffing
    - Sequences of 1s as flag, and then add 0 after each flag within data



Handling Errors

- Two methods
  - Detect errors and correct errors with error codes
  - Detect errors and retransmit frames
- Error correction is much harder than error detection
- Adding redundant check bits to data adds overhead



Hamming Distance

- Minimum number of bits needed to change one valid codeword to another valid codeword
- For a Hamming distance of $d + 1$, up to d errors will be **detected**
- For a Hamming distance of $2d + 1$, up to d errors can be **corrected**

Error Detection

	Description	Hamming Distance
Parity Bit	Add 1 check bit that is sum/XOR of d data bits	2
Internet Checksum	1s complement sum of 16 bit word	2
Cyclic Redundancy Check (CRC)	For n data bits, generate n+k bits that are evenly divisible by C	4

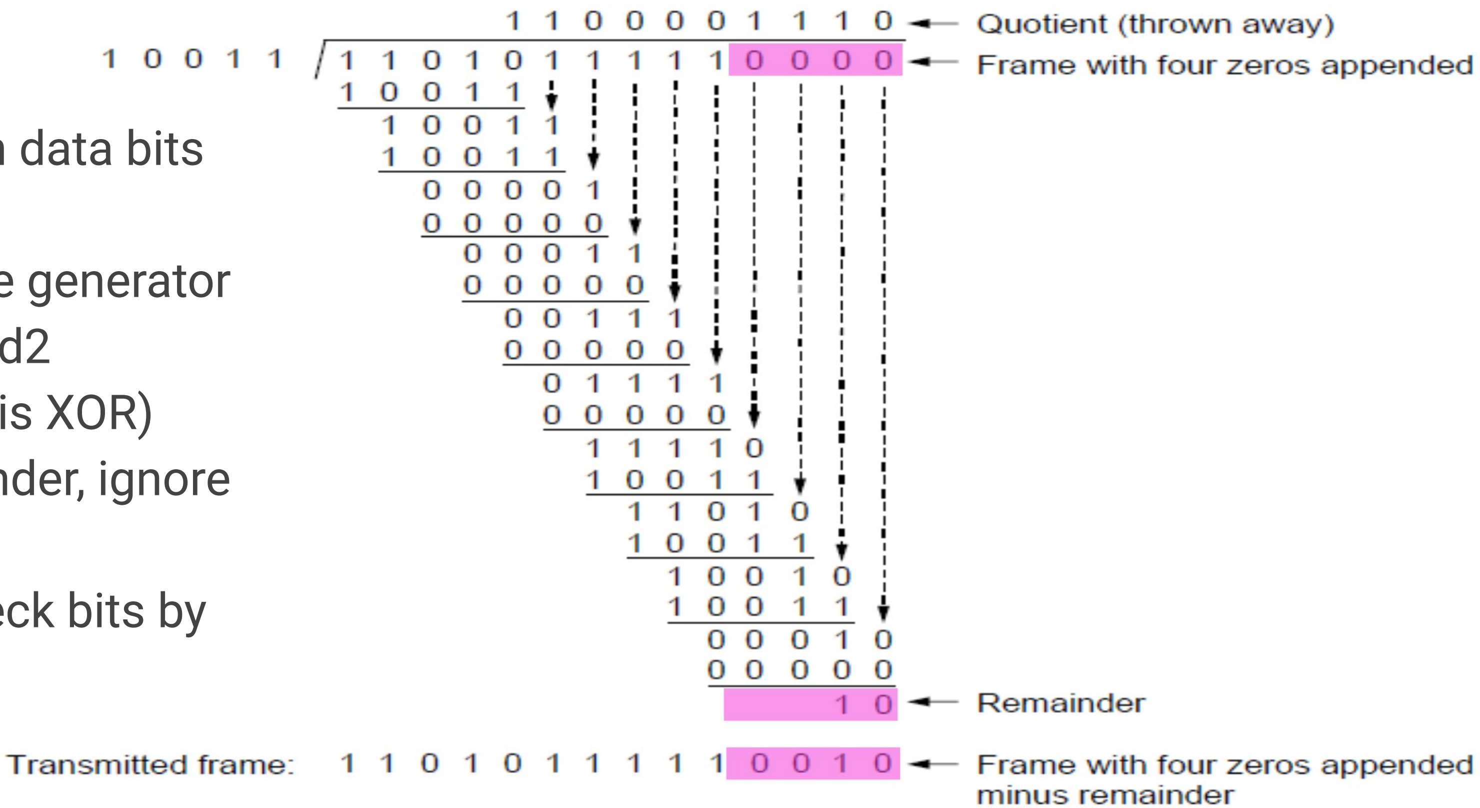
Error Detection - Internet Checksum

- Arrange data in 16-bit words
- Checksum will be non-zero, add
- Add any carryover back to get 16 bits
- Negate (complement) to get sum

```
2188815eeee91 =>
    0002
    1888
    15ee
    ee91
+  (0000)
---------
   11d09
=>
    1d09
+      1
---------
    1d0a
=> e2f5
```

Error Detection - CRC

- Extend the n data bits with k zeros
- Divide by the generator value C (mod2 subtraction is XOR)
- Keep remainder, ignore quotient
- Adjust k check bits by remainder



Error Correction - Hamming Code

Suppose we want to send a message M of 4 bits: **0101**
We add k=3 check bits, because ($n = 2^k - k - 1 = 2^3 - 3 - 1 = 4$)

So, we will have a n+k = 7 bit code, with check bits in positions 1, 2, 4
Each check bit is an XOR of certain positions.

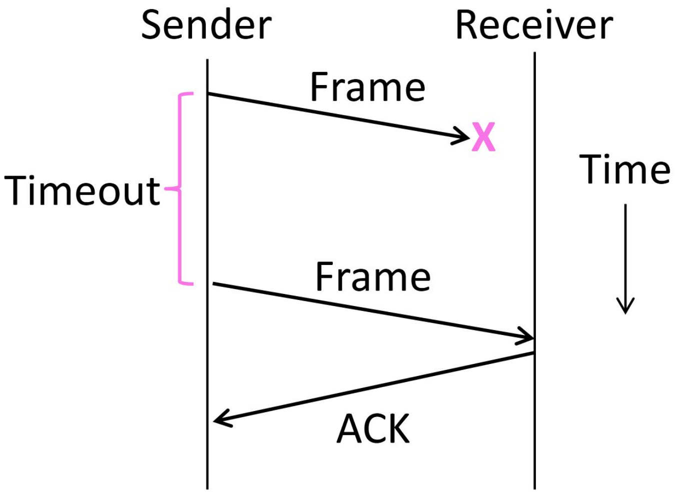
42**1**	4**2**1	**4**21
1 = 0b00**1**	1 = 0b001	1 = 0b001
2 = 0b010	2 = 0b0**1**0	2 = 0b010
3 = 0b01**1**	3 = 0b0**1**1	3 = 0b011
4 = 0b100	4 = 0b100	4 = 0b**1**00
5 = 0b10**1**	5 = 0b101	5 = 0b**1**01
6 = 0b110	6 = 0b1**1**0	6 = 0b**1**10
7 = 0b11**1**	7 = 0b1**1**1	7 = 0b**1**11

0	**1**	**0**	**0**	**1**	**0**	**1**
1	2	3	4	5	6	7

```
p1 = b3+b5+b7 = 0+1+1 = 0
p2 = b3+b6+b7 = 0+0+1 = 1
p4 = b5+b6+b7 = 1+0+1 = 0
```

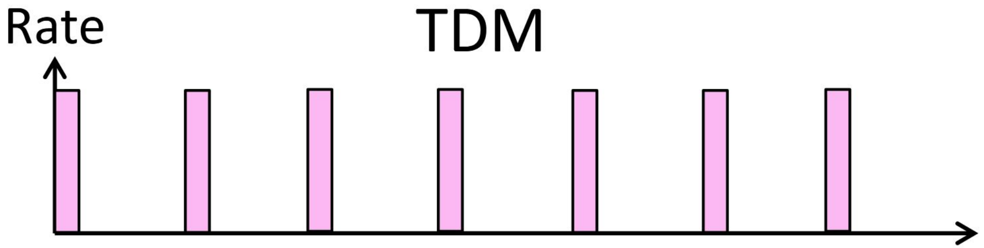
ARQ - Automatic Repeat Request

- ARQ
- Stop-and-wait
- Sliding window

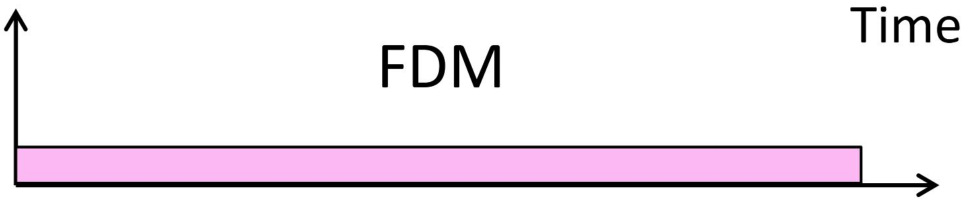


Multiplexing

- Time Division Multiplexing - high rate at some times



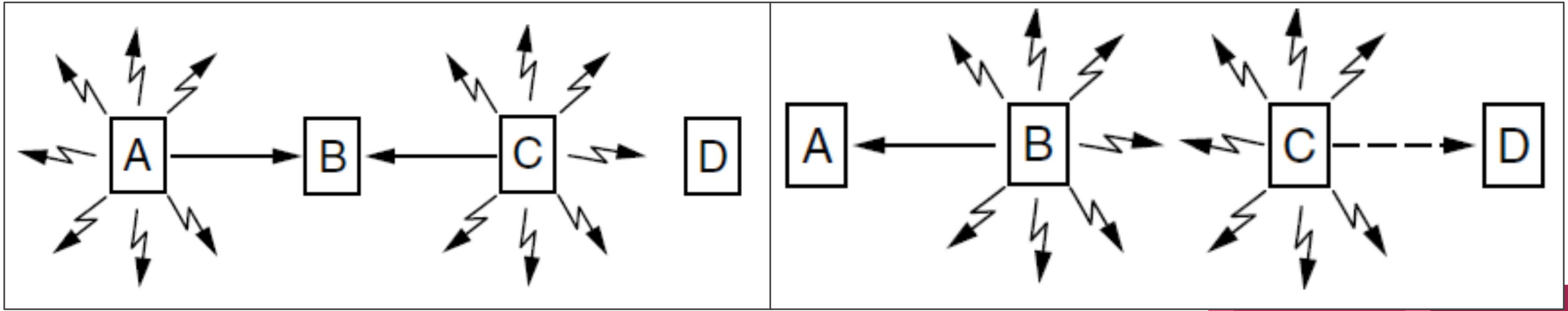
- FDM - low rate all the time



Issues with Wireless

Hidden Terminal Problem: nodes A and C are hidden terminals when sending to B

Exposed Terminal Problem: nodes B and C are exposed terminals when sending to A and D



MACA is a potential solution: Sender sends RTS and receiver sends CTS. Nodes that hear CTS don’t send.

Multiple Access

- ALOHA: Node just sends when it has traffic; if collision happens, wait for a random amount of time and try again.
  - Huge amount of loss under high load
- CSMA (Carrier Sense Multiple Access): Listen before send.
  - Collision is still possible because of delay; good only when BD is small
- CSMA/CD (Carrier Sense Multiple Access with Collision Detection): CSMA + Aborting JAM for the rest of the frame time
  - Minimum frame length of 2D seconds
- CSMA “Persistence”: CSMA + P(send) = 1 / N
  - Reduce the chance of collision
- Binary Exponential Backoff (BEB): Doubles interval for each successive collision
  - Very efficient in practice

Switches

- Backward Learning
  - Learn the sender's port by looking at the packets
- Spanning Tree
  - Elect the root node of the tree (Usually the switch with the lowest address)
  - Grow tree based on the shortest distance from the root
  - Ports not on the spanning tree are turned off

Question Time!

Calculate the latency (from first bit sent to last bit received) for the following:

a) 100 Mbps Ethernet with a single store and forward switch in the path and a packet size of 12,000 bits. Assume that each link introduces a propagation delay of 10 μs and that the switch begins retransmitting immediately after it has finished receiving the packet.
b) Same as (a) but with 3 switches
c) Same as (a), but assume the switch implements "cut through" switching; it is able to begin retransmitting the packet after the first 200 bits have been received.

a) 100 Mbps Ethernet with a single store and forward switch in the path and a packet size of 12,000 bits. Assume that each link introduces a propagation delay of 10 μs and that the switch begins retransmitting immediately after it has finished receiving the packet.

latency = transmission delay + propagation delay = M/R + D

100 Mbps = 10^8 bits/sec 1 bit takes 10^{-8} s to transmit
transmission delay = 12,000 bits / (10^8 bits/sec) = 120 μs
propagation delay = 10 μs

Because there are two links,
latency = (120 μs + 10 μs) * 2 = 260 μs

b) Same as (a) but with 3 switches

Now, there are 3 switches and 4 links.

latency = (120 μs + 10 μs) * 4 = 520 μs

c) Same as (a), but assume the switch implements "cut through" switching; it is able to begin retransmitting the packet after the first 200 bits have been received.

It takes 200 bits / (10^8 bits/sec) = 2 μs for the first 200 bits to get on the first link.
Then, it takes 10 μs for the 200 bits to fully reach the switch.
It takes 120 μs to put all the 12,000 bits on the second link.
It takes 10 μs for all the bits to propagate along the second link.
latency = 2 + 20 + 120 = 142 μs

The last bit still arrives 120 μs after the first bit; the first bit now faces two link delays and one switch delay but never has to wait for the last bit along the way.

3. (15 marks) Consider the two-dimensional parity method for error detection:

		Parity bits
	0101001	1
	1101001	0
	1011110	1
	0001110	1
	0110100	1
	1011111	0
Parity byte	1111011	0

Can all 4 bit errors of this type be detected?

We know that this method cannot catch all 4-bit errors. Now, let's consider in detail several cases concerning where the 4 bit errors occur. First of all, if all four bit errors occur in the parity bits (including the bits in the parity byte), it is clear that all 4-bit error of this type can be detected. How about

a. (5 marks) three bit errors in the parity bits and one in the data,
b. (5 marks) two bit errors in the parity bits and two in the data, and
c. (5 marks) one bit error in the parity bits and three in the data? Explain your answers concisely.

Case (i): This type of error can go undetected, e.g.,

		Parity bits
	010100**1**	**1**
	1101001	0
	1011110	1
	0001110	1
	0110100	1
	1011111	0
Parity byte	111101**1**	**0**

Case (ii): This type of error can go undetected, e.g.,

		Parity bits
	010100**1**	**1**
	110100**1**	**0**
	1011110	1
	0001110	1
	0110100	1
	1011111	0
Parity byte	1111011	0

Case (iii): All errors of this type can be detected, because the only parity bit error can go undetected by having a data bit error in either the same column or row as the parity bit in error. But, the other two data bit errors cannot go undetected.

Suppose we send a message of 11 bits and add 4 check bits at the end.

Here are the bits that each check bit covers:

```
1 =>  1, 3, 5, 7, 9,11,13,15
2 =>  2, 3, 6, 7,10,11,14,15
4 =>  4, 5, 6, 7,12,13,14,15
8 =>  8, 9,10,11,12,13,14,15
```

Say that the receiver received the following message:

0	0	0	0	1	0	1	0	1	0	0	1	0	0	0
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

What is the syndrome? Which bit is wrong?

0	0	0	0	1	0	1	0	1	0	0	1	0	0	0
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

The syndrome is calculated as follows:

p1=(0+0+1+1+1+0+0+0) mod 2=1
p2=(0+0+0+1+0+0+0+0) mod 2=1
p4=(0+1+0+1+1+0+0+0) mod 2=1
p8=(0+1+0+0+1+0+0+0) mod 2=0

```
1 =>  1, 3, 5, 7, 9,11,13,15
2 =>  2, 3, 6, 7,10,11,14,15
4 =>  4, 5, 6, 7,12,13,14,15
8 =>  8, 9,10,11,12,13,14,15
```

=> syndrome=p8p4p2p1=0111

So, the bit flipped is bit 7.