CSE 461: Introduction to Computer Communications Networks
Winter 2010

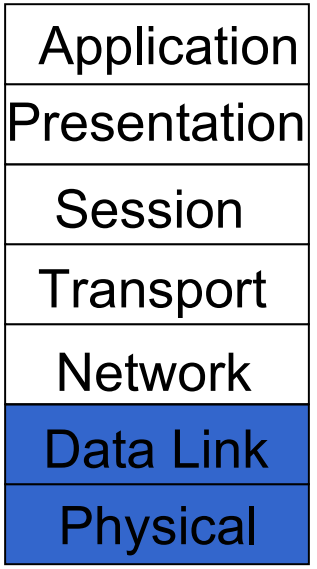
Module 3
Direct Link Networks – Part A

John Zahorjan
zahorjan@cs.washington.edu
534 Allen Center

This Module's Topics

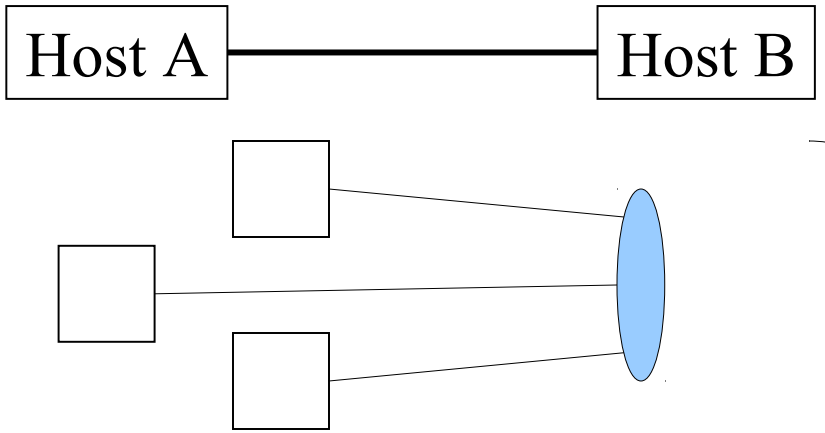
Overview of Computer Networking

1. Overview – Scope of today's discussion
2. Encoding / Framing / Error Detection
3. Reliable Transmission



Direct Link Networks

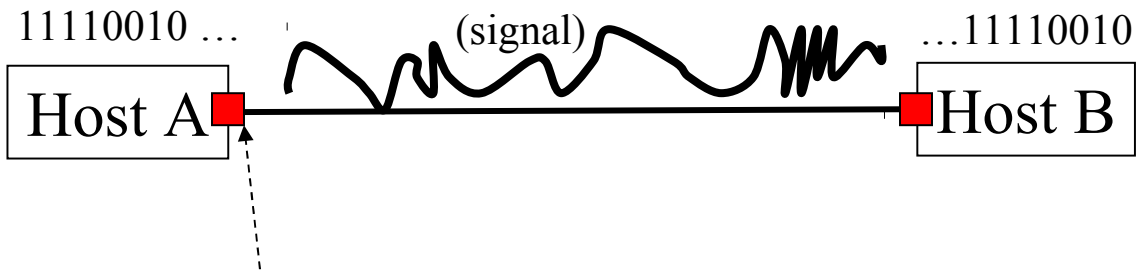
"Direct link" ⇒ no switching/routing (or addressing)



Point-to-point

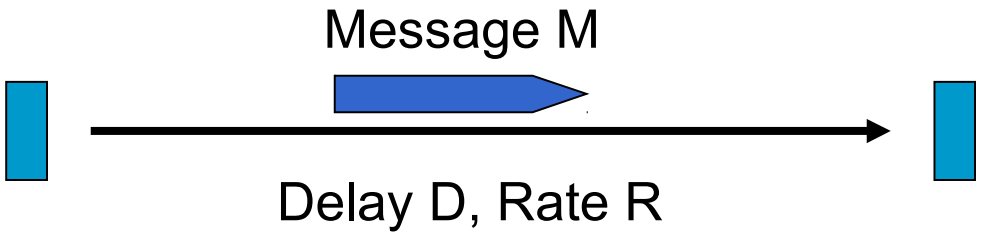
Broadcast / shared

Relationship to the hardware



What really happens

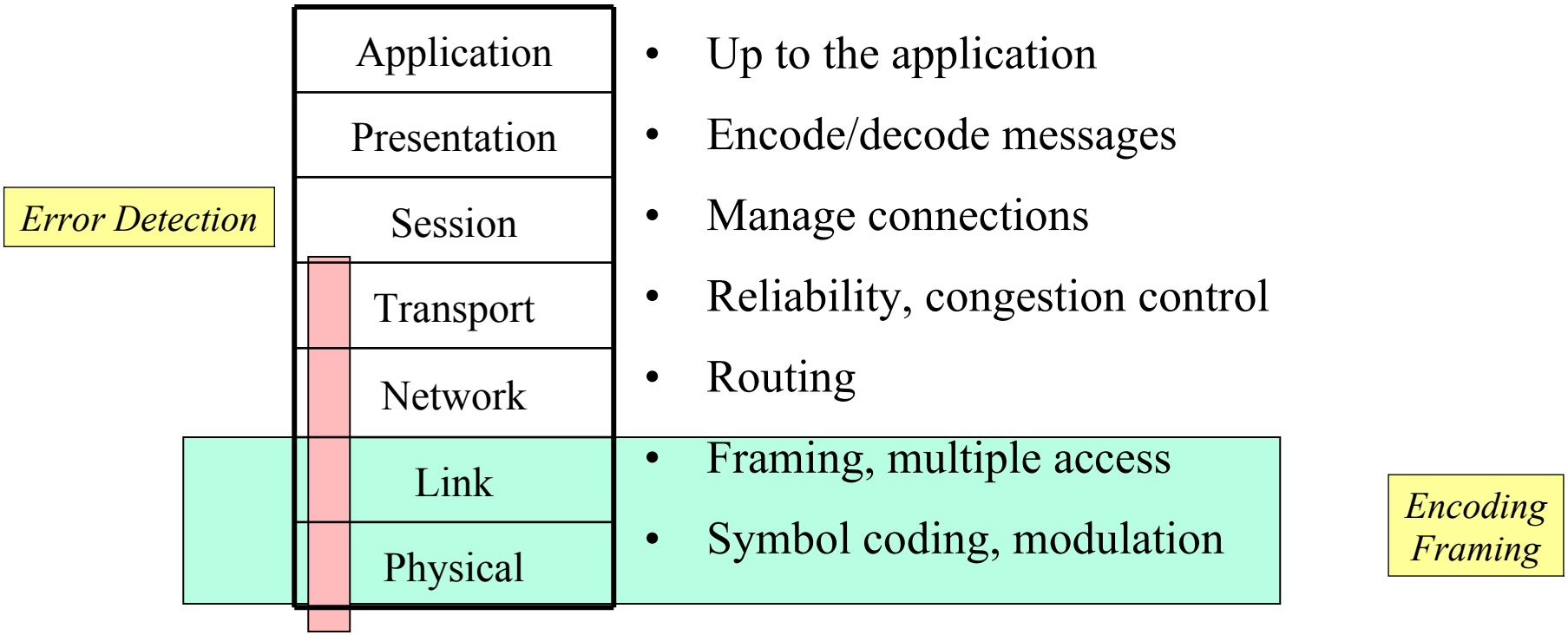
Network interface cards (NICs)
(also called "network adaptors")



Abstract link for our purposes

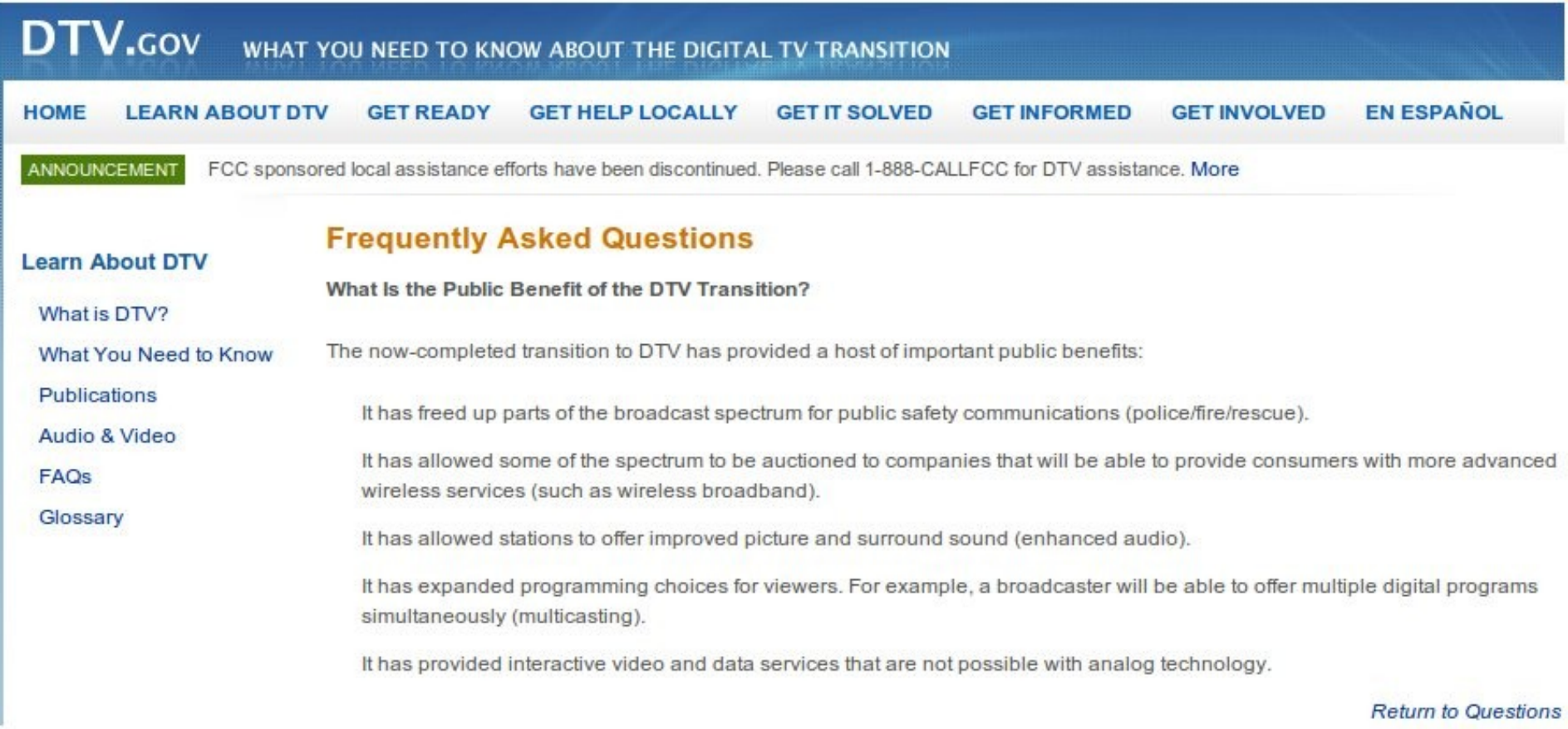
For now, "messages" are bits.

Relationship to the Protocol Stack



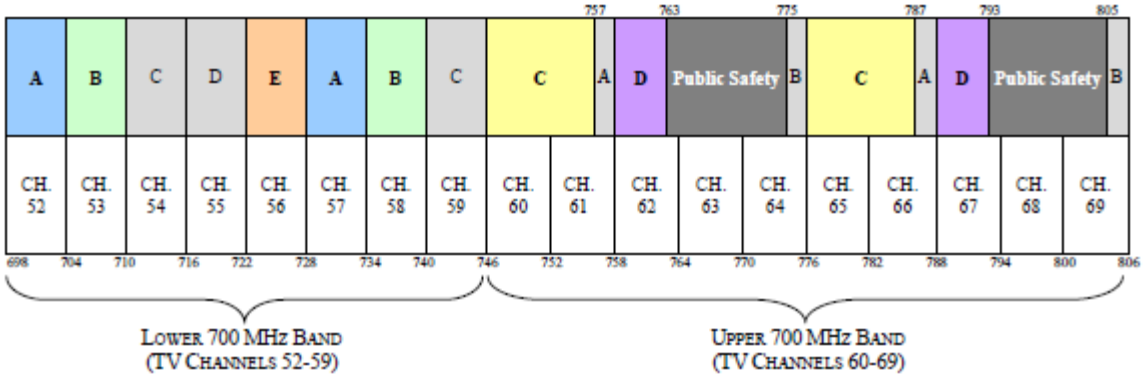
Remember, this is an idealization of what actually goes on (and the organization of the book is explicitly non-layerist).

Links Are Valuable



FCC Auction 73

Revised 700 MHz Band Plan for Commercial Services



Block	Frequencies (MHz)	Bandwidth	Pairing	Area Type	Licenses
A	698-704, 728-734	12 MHz	2 x 6 MHz	EA	176
B	704-710, 734-740	12 MHz	2 x 6 MHz	CMA	734
C	710-716, 740-746	12 MHz	2 x 6 MHz	CMA	734
D	716-722	6 MHz	unpaired	EAG	6
E	722-728	6 MHz	unpaired	EA	176
C	746-757, 776-787	22 MHz	2 x 11 MHz	REAG	12
A	757-758, 787-788	2 MHz	2 x 1 MHz	MEA	52
D	758-763, 788-793	10 MHz	2 x 5 MHz	Nationwide	1 *
B	775-776, 805-806	2 MHz	2 x 1 MHz	MEA	52

* Subject to conditions respecting a public/private partnership.

The blocks shaded above in gray (Lower 700 MHz Band C and D Blocks and Upper 700 MHz Band A and B Blocks) were auctioned prior to Auction 73.

Links Are Valuable

The New York Times — March 21, 2008

And the Winners Are . . .

The government made $19.1 billion in its auction of wireless spectrum to 101 companies. The big spenders were established cellphone carriers, although a satellite TV company was a winner.

COMPANY	VALUE OF WINNING BIDS
Cellco Partnership (**Verizon Wireless**)	**$9,363,160,000**
AT&T Mobility Spectrum	**6,636,658,000**
Frontier Wireless (**Dish Network**)	**711,871,000**
Qualcomm	**558,142,000**
King Street Wireless (**US Cellular**)	**400,638,000**
MetroPCS 700 MHz	313,267,000
Cox Wireless	304,633,000
Cellular South Licenses	191,533,000
CenturyTel Broadband Wireless	148,964,000
Vulcan Spectrum (backed by Paul Allen)	112,793,000
Continuum 700	88,179,000
Cavalier Wireless	61,803,000
Puerto Rico Telephone Co.	31,402,000
Triad 700	22,694,000
McBride Spectrum Partners	8,490,000

Sources: Federal Communications Commission; Goldman Sachs

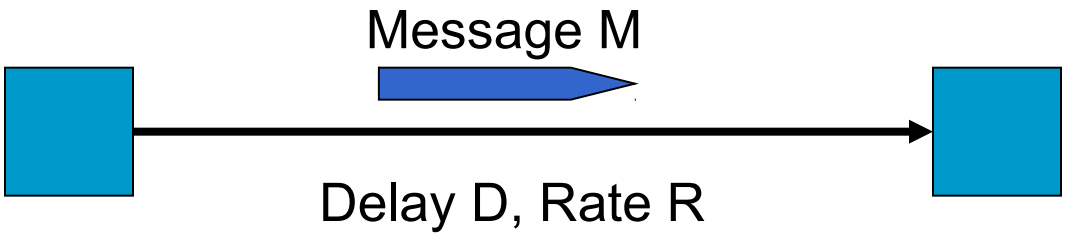
THE NEW YORK TIMES

Model of a Link

- Abstract model is typically all we will need
- Other parameters that are important:
  - The kind and frequency of errors (bit error rate, BER)
  - Whether the media is broadcast or not

Message Latency

- How long does it take to send a message?



- Two terms:
  - Propagation delay = distance / speed of signal in media
    - How quickly a message travels over the wire
    - 2/3c for copper wire
  - Transmission delay = message (bits) / rate (bps)
    - How quickly you can inject the message onto the wire
  - Propagation delay tells you when the FIRST bit arrives, Transmission delay tells you when the LAST bit arrives.

One-way Latency

Dialup with a modem:

- D = 10ms, R = 56Kbps, M = 1024 bytes
- Latency = 10ms + (1024 x 8)/(56 x 1024) sec = 153ms!

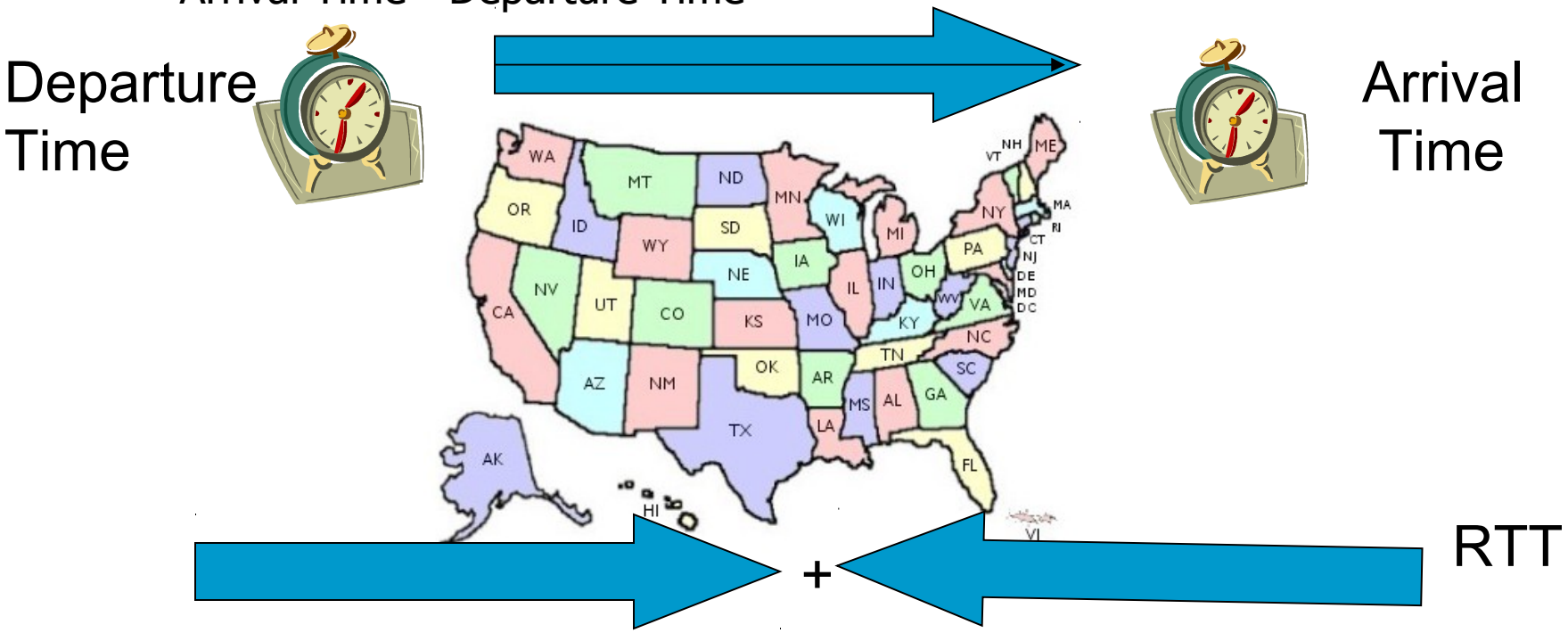
Cross-country with T3 (45Mbps) line:

- D = 50ms, R = 45Mbps, M = 1024 bytes
- Latency = 50ms + (1024 x 8) / (45 x 1024*1024) sec = 50ms!

- Either a slow link or long wire makes for large latency

Latency and RTT

- Latency is typically the one way delay over a link
  - Arrival Time - Departure Time



- The round trip time (RTT) is twice the one way delay (assuming symmetry)
  - Measure of how long to signal and get a response

Throughput

- Measure of system's ability to "pump out" data
  - NOT the same as bandwidth
- Throughput = Transfer Size / Transfer Time
  - E.g., "I transferred 1000 bytes in 1 second on a 100Mb/s link"
    - BW?
    - Throughput?
- Transfer Time = SUM OF
  - Time to get started shipping the bits
  - Time to ship the bits
  - Time to get a response if necessary

Messages Occupy Space On the Wire

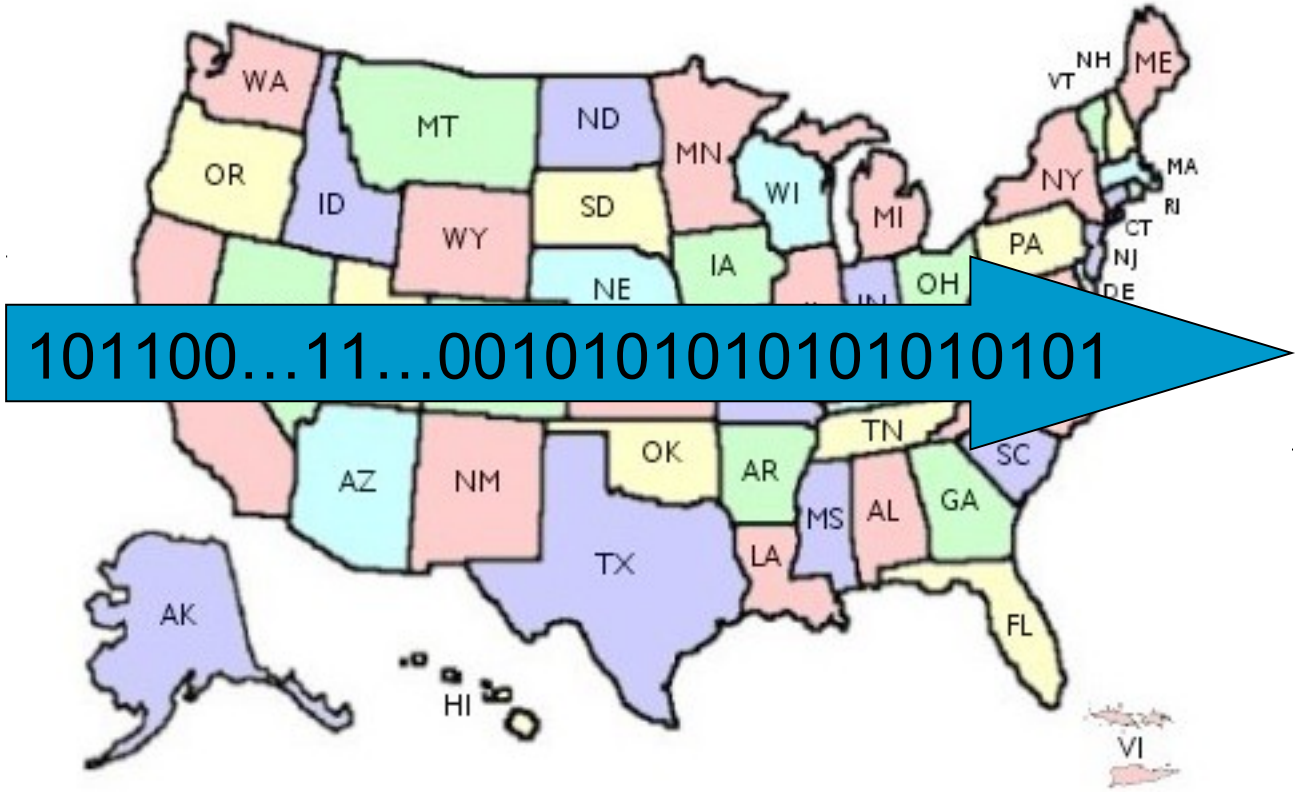
- Consider a 1b/s network.

- Suppose latency is 16 seconds.
  - How many bits can the network "store"
  - This is the BANDWIDTH-DELAY product (BD)
  - Measure of "data in flight."
  - 1b/s * 16s = 16b

- Tells us how much data can be sent before a receiver sees any of it.
  - Twice BD tells us how much data we could send before hearing back from the receiver something related to the first bit sent.
  - What are the implications of high B.D.?

A More Realistic Example

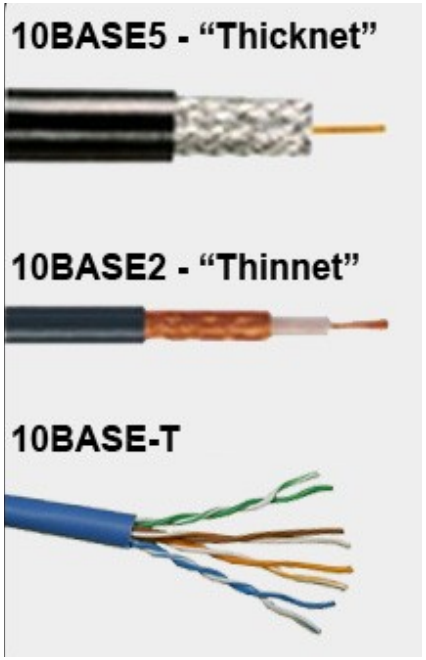
BD = 50ms * 45Mbps = 2.25 * 10^6 = 280KB



We'll see why this is important when we learn about TCP

Model of a wire

- Frequencies beyond cutoff highly attenuated
  - Bandwidth = passband (Hz)
- Signal also subject to:
  - Attenuation (repeaters)
  - Distortion (frequency and delay)
  - Noise (thermal, crosstalk, impulse)



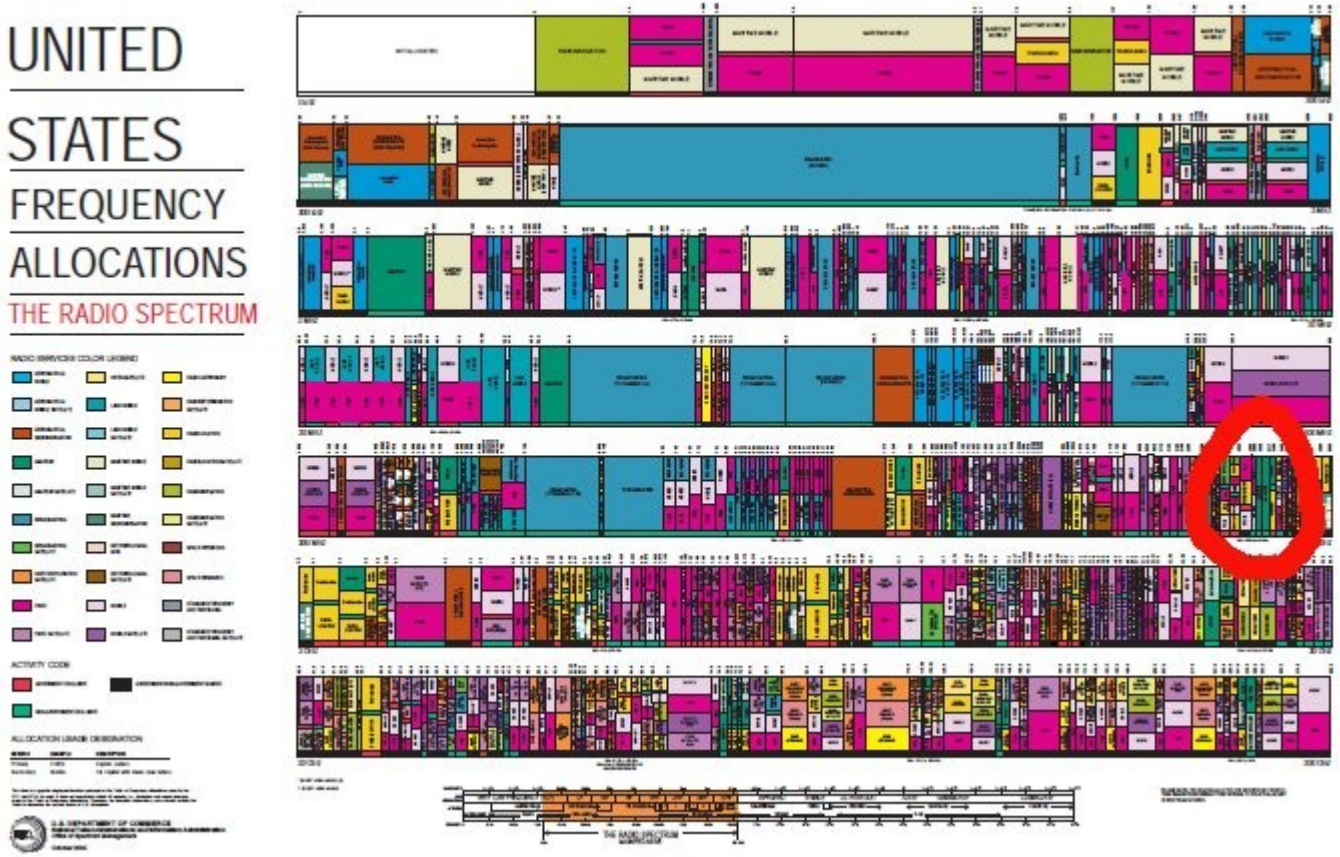
Cat 5e, 6, and 7 for GigE

EE: Bandwidth = width of frequency passband, measured in Hz
CS: Bandwidth = information carrying capacity, measured in bits/sec

Model of the Air

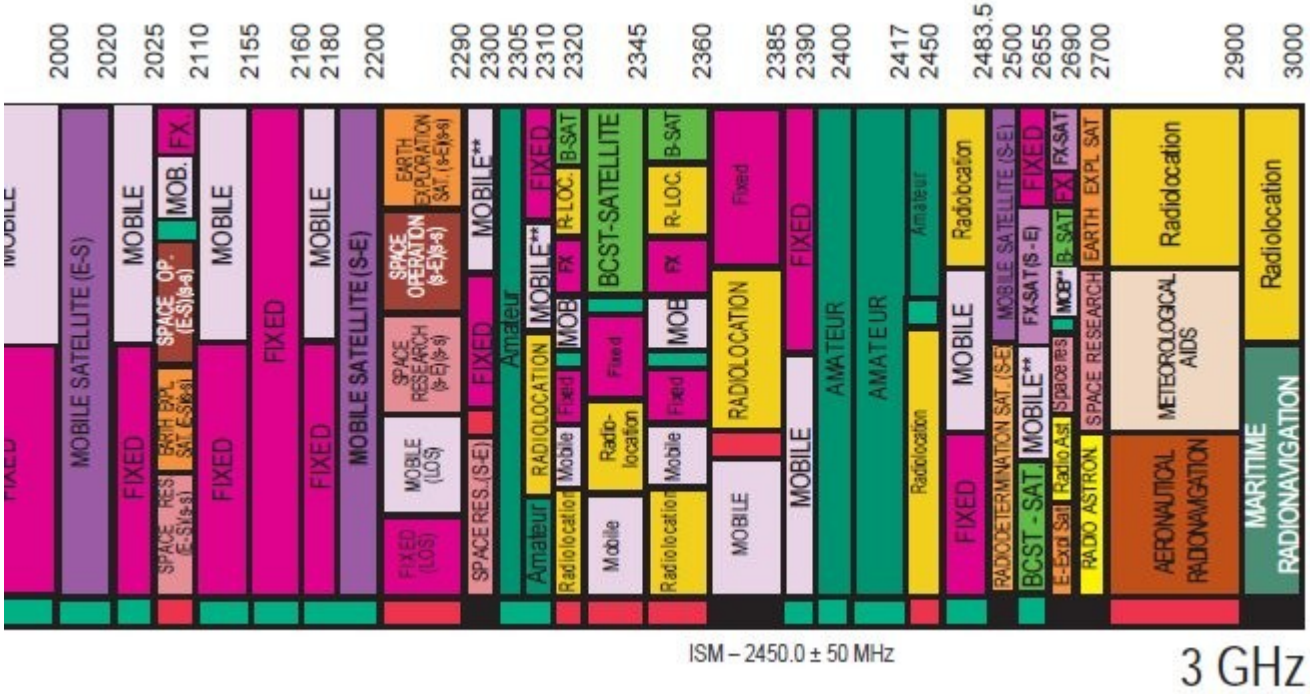
- Frequencies beyond and below cutoff highly attenuated
  - You're required to keep to your own spectrum
- Signal also subject to:
  - Attenuation (nominally distance^2)
  - Distortion (multipath, fading, shadowing)
  - Noise (thermal, interference)
- You're also subject to transmit power restrictions
- All of this makes wireless a more error prone environment

U.S. Frequency Allocation

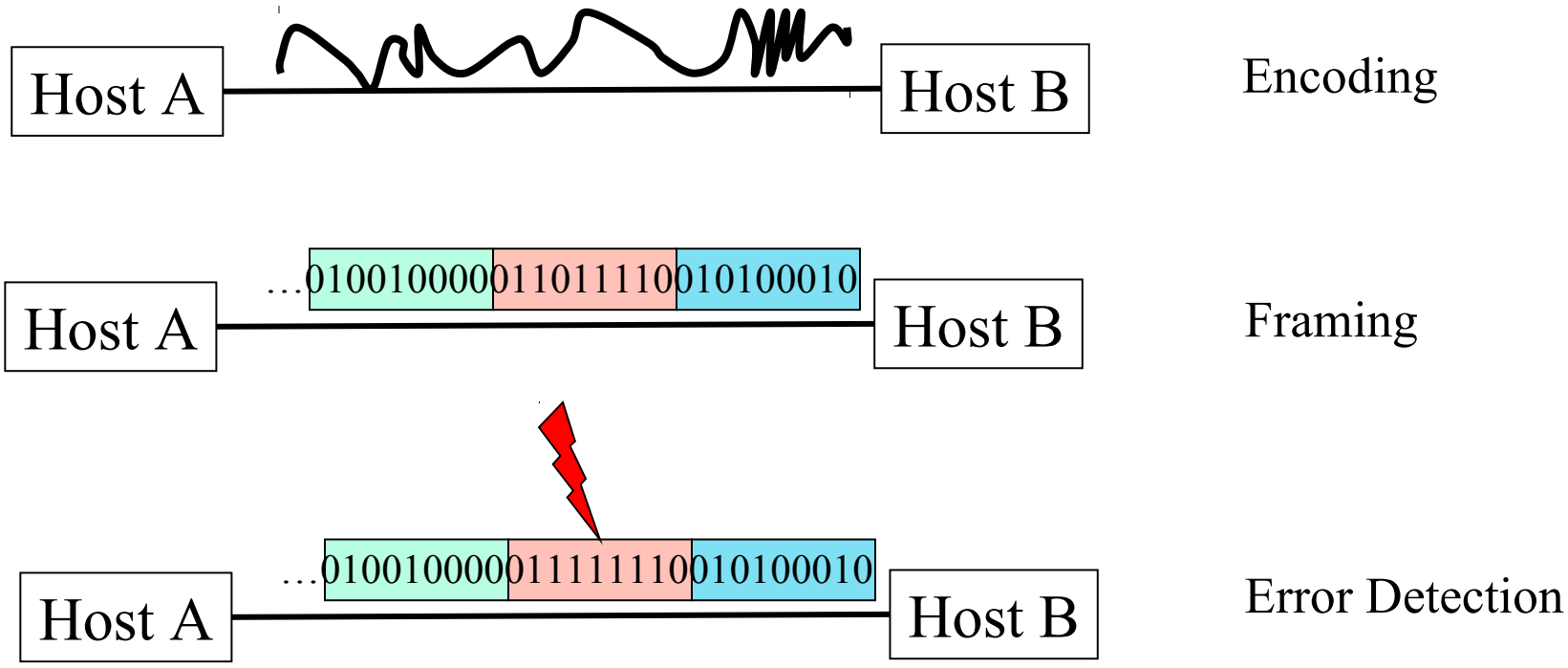


http://www.ntia.doc.gov/osmhome/allochrt.pdf

802.11 a/b/g (/n)


2000
2020
2025
2110
2155
2160
2180
2200
2290
2300
2305
2310
2320
2345
2360
2385
2390
2400
2417
2450
2483.5
2500
2655
2690
2700
2900
3000
MOBILE SATELLITE (E-S)
FIXED
MOBILE
SPACE RES. (E-S)(s-s)
EARTH EXPL. SAT. (E-S)(s-s)
SPACE OP. (E-S)(s-s)
MOB.
FX
FIXED
MOBILE
FIXED
MOBILE
FIXED
MOBILE
MOBILE SATELLITE (S-E)
FIXED (LOS)
MOBILE (LOS)
SPACE RESEARCH (S-E)(s-s)
SPACE OPERATION (S-E)(s-s)
EARTH EXPLORATION SAT. (S-E)(s-s)
SPACE RES.(S-E)
FIXED
MOBILE**
Amateur
Amateur
RADIOLOCATION
MOBILE**
FIXED
Radiolocation
Mobile
Fixed
MOB
FX
R-LOC.
B-SAT
Mobile
Radio-location
Fixed
BCST-SATELLITE
Radiolocation
Mobile
Fixed
MOB
FX
R-LOC.
B-SAT
MOBILE
RADIOLOCATION
Fixed
MOBILE
FIXED
AMATEUR
AMATEUR
Radiolocation
Amateur
FIXED
MOBILE
Radiolocation
RADIODETERMINATION SAT. (S-E)
MOBILE SATELLITE (S-E)
BCST - SAT.
MOBILE**
FX-SAT (S - E)
FIXED
E-Exp Sat
RadioAst.
Spacetrks
MOB*
R-SAT
FX
FX-SAT
RADIO ASTRON.
SPACE RESEARCH
EARTH EXPL SAT
AERONAUTICAL RADIONAVIGATION
METEOROLOGICAL AIDS
Radiolocation
MARITIME RADIONAVIGATION
Radiolocation
ISM – 2450.0 ± 50 MHz
3 GHz

Direct Link Networks



Encoding, Symbols, and Bits

- JZ waves

- Baud rate is symbols; bit rate is information

Fundamental Limits

- Nyquist rate on maximum symbols/second:

  $R < 2B$

  - Channel is bandwidth-limited to B

- Shannon capacity of a channel:

  $C = B \cdot \log_2 (1 + S / N)$

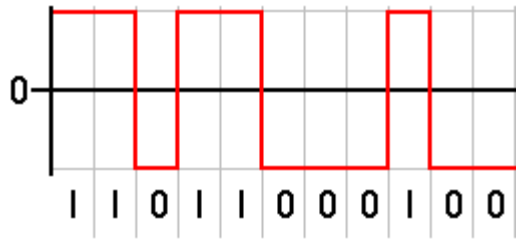
  - N is additive-white-Gaussian-noise (AWGN)

- Bandwidth and signal-to-noise ratio define fundamental limits

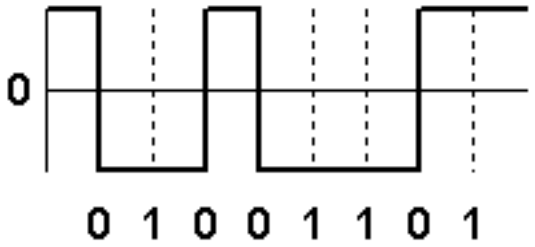
Encoding

- Modulate something – amplitude, frequency, phase
- A key issue is clocking
  - Higher transmission rates require better synch

- Some example encodings (thanks, *wikipedia)*:



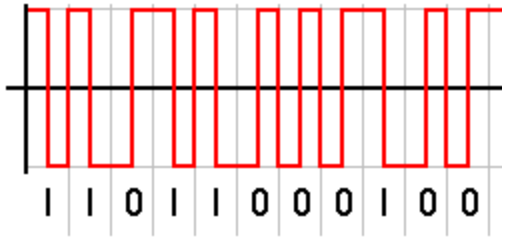
NRZ
(RS-232)



NRZI
(CDs, USB, Fast Ethernet)

Encoding: Self-Clocking

- Receiver can derive clock from the data signal
- Example 1:



Manchester
(10Mbps Ethernet)

- Example 2: Use NRZI, but make sure there are transitions
  - 4B/5B multi-level transition (MLT)
    - 100Mbps Ethernet, with 3 levels of signal
  - 8B/10B MLT
    - 1000Mbps Ethernet, with 5 levels of signal
  - (MLT is used to limit the required signal bandwidth to what can be carried on cheap, CAT 5 cable (100MHz).)

Separate Clock Distribution

- Self-clocking consumes bandwidth
  - Manchester: two transitions per bit
  - 4B/5B and 8B/10B: overhead of additional bits

- Alternative: send explicit clock
  - SONET (Synchronous Optical NETwork)
    - Clock can be carried explicitly from one network element to another
    - Nodes can all use clock from GPS
    - Various fallbacks

Table A: Stratum Clock Requirements and Hierarchy

Stratum	Accuracy/Adjust Range	Pull-In-Range	Stability	Time To First Frame Slip *
1	1×10^{-11}	N/A	N/A	72 Days
2	1.6×10^{-8}	Must be capable of synchronizing to clock with accuracy of $+/-1.6 \times 10^{-8}$	1×10^{-10}/day	7 Days
3E	1.0×10^{-6}	Must be capable of synchronizing to clock with accuracy of $+/-4.6 \times 10^{-6}$	1×10^{-8}/day	3.5 Hours
3	4.6×10^{-6}	Must be capable of synchronizing to clock with accuracy of $+/-4.6 \times 10^{-6}$	3.7×10^{-7}/day	6 Minutes (255 in 24 Hrs)
4E	32×10^{-6}	Must be capable of synchronizing to clock with accuracy of $+/-32 \times 10^{-6}$	Same as Accuracy	Not Yet Specified
4	32×10^{-6}	Must be capable of synchronizing to clock with accuracy of $+/-32 \times 10^{-6}$	Same as Accuracy	N/A

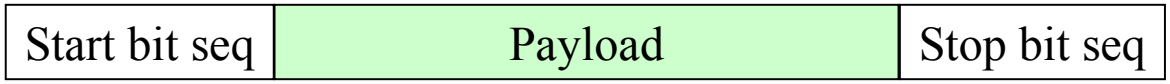
Framing

- Need to send message, not just bits
  - Requires that we synchronize on the start of message reception at the far end of the link
  - Complete Link layer messages are called frames

- Common approach: Sentinels
  - Look for special sequence that marks start of frame, e.g., preamble in 802.11, 0x7E in PPP
  - May escape or "stuff" this code within the data region, e.g., PPP
    - Like a C compiler: A quotation mark (") is a string sentinel, so (\") means (")
  - May give length of frame with header, e.g., SONET, 802.11

Framing (cont.)

- The generic view



- Because the payload may contain the start or stop sequence, may have to "stuff" payload at sender, and unstuff at receiver
  - Something like putting a quote inside a quoted string in a programming language
    - Suppose start bit sequence is 0x7E.
    - Replace 0x7E in payload with 0x7D 0x5E
    - Replace 0x7D in payload with 0x7D 0x5D
    - At receiver, 0x7D 0x5E replaced with 0x7E
- We'll see more frame formats when we look at specific link level protocols in a bit…

Problem: Transmission Errors
Solution: Redundancy

- Noise can flip some of the bits we receive
  - 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
- Questions we'll delay for a bit:
  - What should happen if an uncorrectable error is detected?
  - Which layer(s) should do whatever it is?

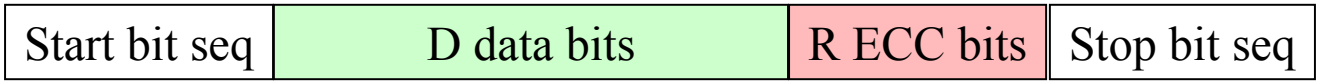
Patterns of Errors

- Q: Suppose you expect a bit error rate of about 1 bit per 1000 sent. What fraction of packets would be corrupted if they were 1000 bits long (and you could detect all errors but correct none)?

- A: It depends on the pattern of errors
  - Bit errors occur at random
    - Packet error rate is about $1 - 0.999^{1000} = 63\%$
  - Errors occur in bursts, e.g., 100 consecutive bits every 100,000 bits
    - Packet error rate ≤ 2%

Error Detection/Correction Codes

- Detection/correction schemes are characterized in two ways:
  - Overhead: ratio of total bits sent to data bits, minus 1
    - Example: 1000 data bits + 100 code bits = 10% overhead
  - The errors they detect/correct
    - E.g., all single-bit errors, all bursts of fewer than 3 bits, etc.
- A scheme maps D bits of data into D+R bits – i.e., it uses only 2^D distinct bit strings of the 2^{D+R} possible.



- The sender computes the ECC bits based on the data.
- The receiver also computes ECC bits for the data it receives and compares them with the ECC bits it received.
  - Detection occurs when what the receiver computed and received don't match
  - That is, detection occurs when the D+R total bits are not one of the 2^D messages valid using the code

The Hamming Distance

- The Hamming distance of a code is the smallest number of bit flips that turn any one codeword into another
  - e.g, code 000 for 0, 111 for 1, Hamming distance is 3
- For code with distance d+1:
  - d bit 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 → 000

Specific Schemes

- We'll briefly touch on the three schemes mentioned in the book
  - They're organized from least to most expensive to compute

- Scheme 1: parity
  - A single parity bit is associated with each K bits of the data, for some K. It is set so that the XOR of the data bits + the parity bit = 0 (for even parity)
    - Example: K=8, one parity bit per byte
      - Detects all odd numbers of errored bits
    - Example: 2-dimensional parity: one parity bit for each bit in a byte, another for each of the eight bit positions in 8 consecutive bytes
      - Detects all 1-, 2-, and 3- bit errors, plus many >3-bit errors

2-d parity example

0101001	1
1101001	0
1011110	1
0001110	1
0110100	1
1011111	0
1111011	0

Specific Schemes

- Scheme 2: checksum
  - General idea: Sum successive blocks of K-bits of the data, as though they were integers
  - Internet checksum: K=16, use 1's-complement arithmetic, take 1's complement of result as checksum
    - Example: data is 01 00 F2 03 F4 F5 F6 F7
      - 0100 + F203 = [0] F303
      - F303 + F4F5 = [1] E7F8 = E7F9
      - E7F9 + F6F7 = [1] DEF0 = DEF1
      - Checksum is 1's complement of DEF1: 210E
      - Transmit 01 00 F2 03 F4 F5 F6 F7 21 0E
    - Why use 1's-complement is a bit arcane (e.g., endian-ness of machine doesn't matter), and not terribly crucial

Specific Schemes

- CRCs (Cyclic Redundancy Check)
  - Stronger protection than checksums
    - Used widely in practice, e.g., Ethernet CRC-32
    - Implemented in hardware (XORs and shifts)

- Based on mathematics of finite fields
  - "numbers" correspond to polynomials, use modulo arithmetic
  - e.g, interpret 10011010 as $x^7 + x^4 + x^3 + x^1$

- Algorithm: Given n bits of data, generate a k bit check sequence that gives a combined n + k bits that are divisible by a chosen divisor C(x)

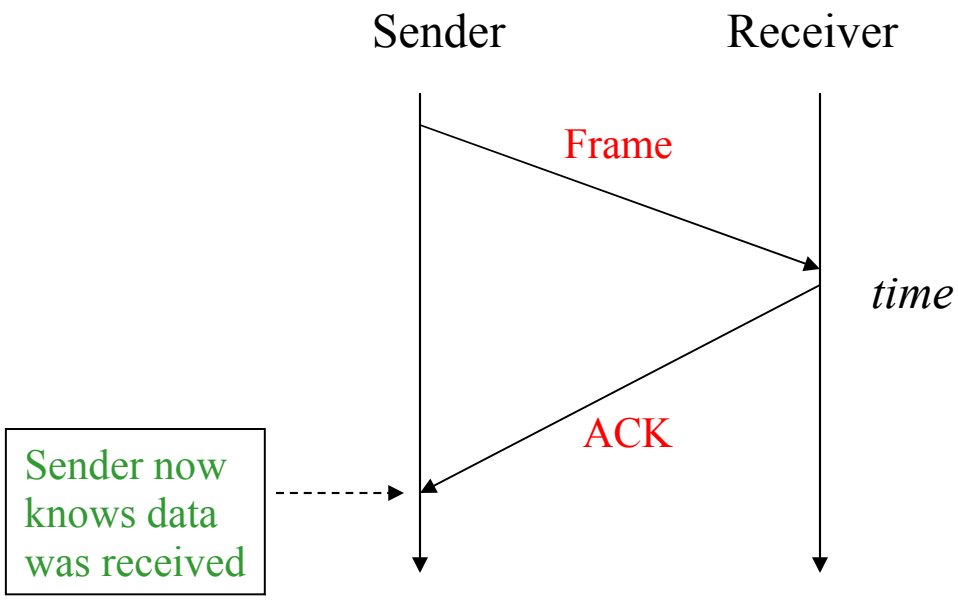
How is C(x) Chosen?

- 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
  - Ethernet CRC-32: 10000010011000001000111011011011 1

Reliable Transmission

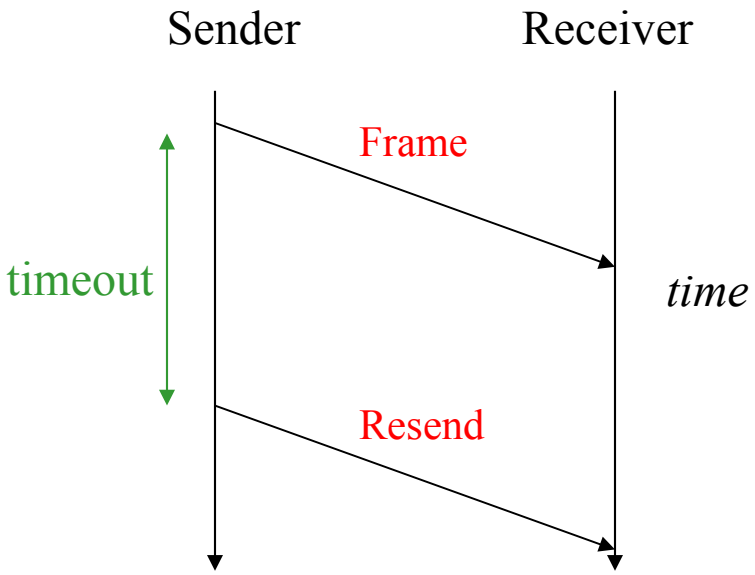
- Because there may be uncorrectable errors (no matter what ECC scheme is used), how can the sender be sure that the receiver got the data?
  - The sender must receive an acknowledgement (ACK) from the sender



What if noACK is received?

Timeouts / Automatic Repeat Request (ARQ)

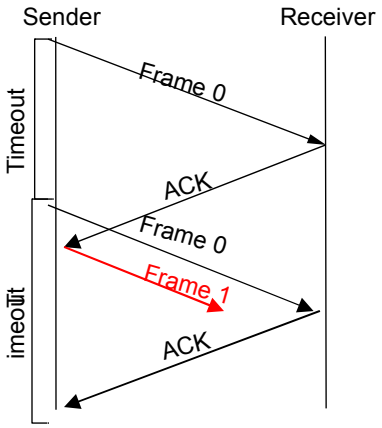
- If no ACK comes back, the sender must re-send the data (ARQ)
  - When is the sender sure that no ACK is coming back?
    - Because as a practical matter delays are very difficult to bound, in some sense it can never be sure
    - Sender chooses some reasonable timeout – if the ACK isn't back in that much time, it assumes it will never see an ACK, and re-sends



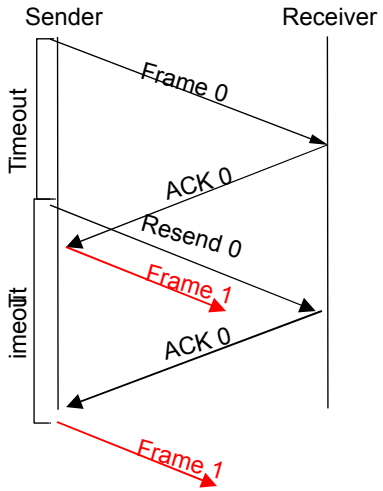
What if original frame arrived, but ACK was lost?

Stop & wait sequence numbers

- Sequence numbers enable the receiver to discard duplicates
- ACKs must carry sequence number info as well



The Problem Scenario



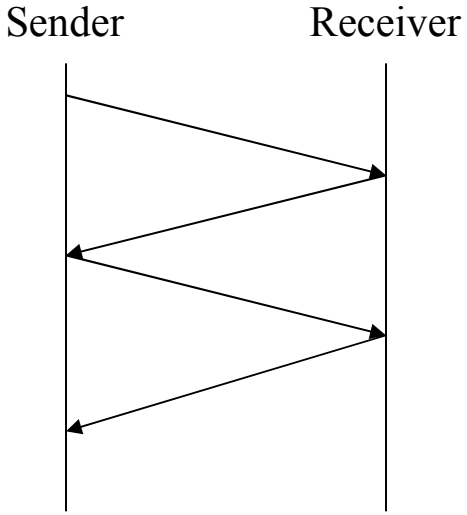
The Solution

- Stop & wait allows one outstanding frame, requires two distinct sequence numbers

Duplicate Detection: Sequence Numbers

- So that the receiver can detect (and discard) duplicates, distinct frames are given distinct sequence numbers
  - E.g., 0, 1, 2, 3, …
- When a frame is re-sent, it is re-sent with the same sequence number as the original
- The receiver keeps some information about what sequence numbers it has seen, and discards arriving packets that are duplicates

Stop-and-Wait Protocol



Here's what it looks like when things are going well (no transmission errors).

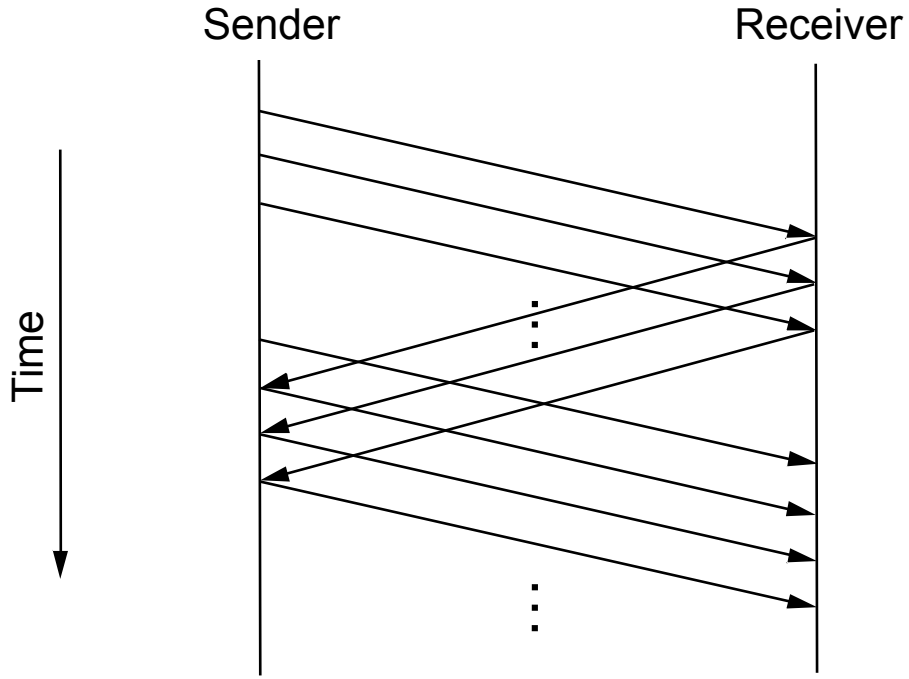
- Sender doesn't send next packet until he's sure receiver has last packet
- The packet/ACK sequence enables reliability
- Sequence numbers help avoid problem of duplicate packets

Problem with Stop-And-Wait: Performance

- Problem: "keeping the pipe full"
  - If the bandwidth-delay product is much larger than a packet size, the sender will be unable to keep the link busy
- Example
  - 1.5Mbps link x 45ms RTT = 67.5Kb (8KB)
  - 1KB frames implies 1/8th link utilization
- Solution: allow multiple frames "in flight"

Solution: Allow Multiple Frames in Flight

- This is a form of pipelining



Flow Control

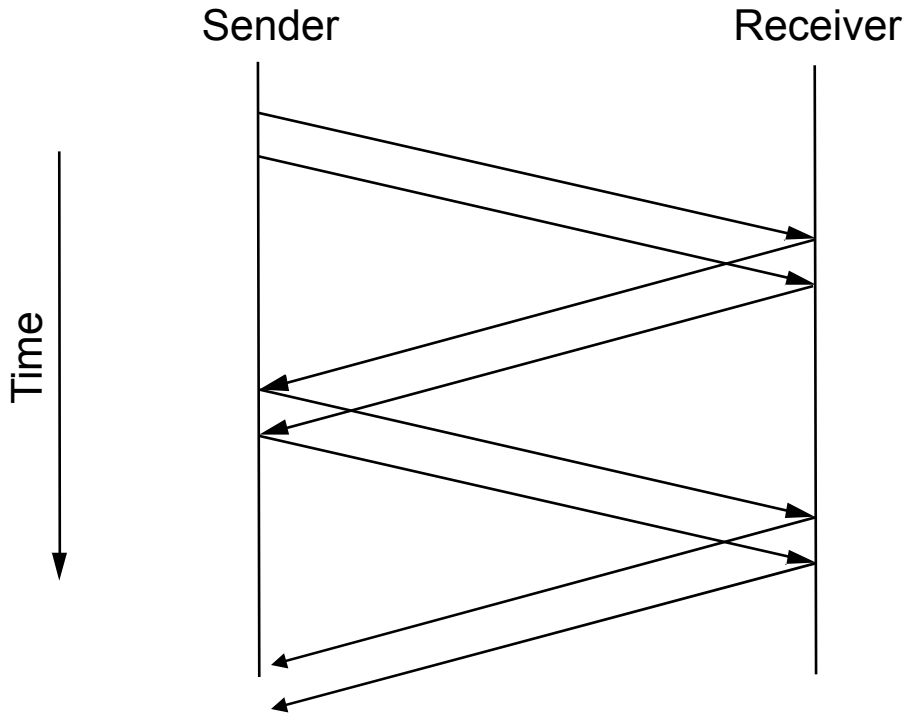
- Why can't we allow the sender to send as fast as it can, timing out and re-sending each frame as necessary?

- Flow control:
  - Receiver needs to buffer data until it can be delivered to higher layers
    - If the sender is much faster than the receiver, it will overwhelm it, causing the receiver to run out of buffer space
  - Additionally, if a frame is lost, the receiver will receive frames "out of order". It wants to buffer those frames to avoid retransmission, but cannot deliver them to the client until the missing frame is re-sent and received
  - Flow control is the notion that the receiver must be able to control the rate at which the sender is thrusting frames at it

- A common, important approach to flow control is the *sliding window protocol*

Sliding Window Protocol

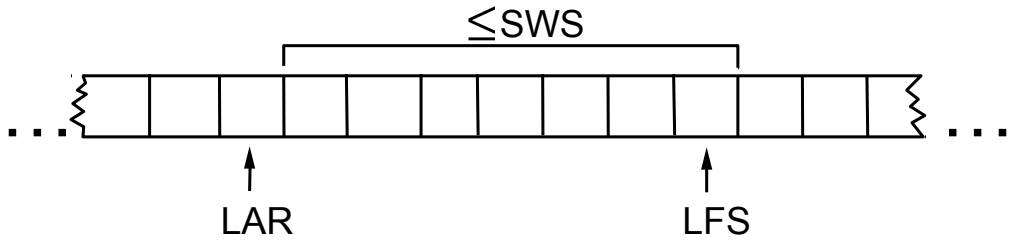
- There is some maximum number of un-ACK'ed frames the sender is allowed to have in flight
  - We call this "the window size"
  - Example: window size = 2



Once the window is full, each ACK'ed frame allows the sender to send one more frame

Sliding Window: Sender

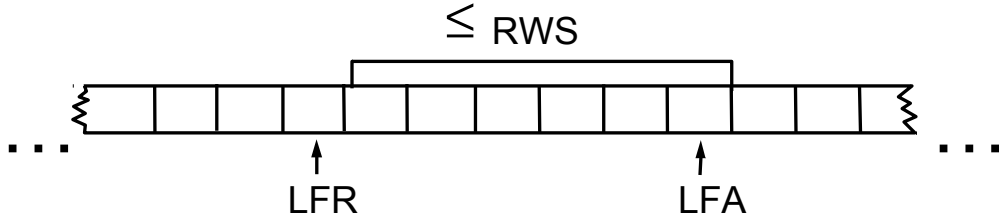
- Assign sequence number to each frame (`SeqNum`)
- Maintain three state variables:
  - send window size (`SWS`)
  - last acknowledgment received (`LAR`)
  - last frame sent (`LFS`)
- Maintain invariant: `LFS - LAR <= SWS`



- Advance `LAR` when ACK arrives
- Buffer up to `SWS` frames

Sliding Window: Receiver

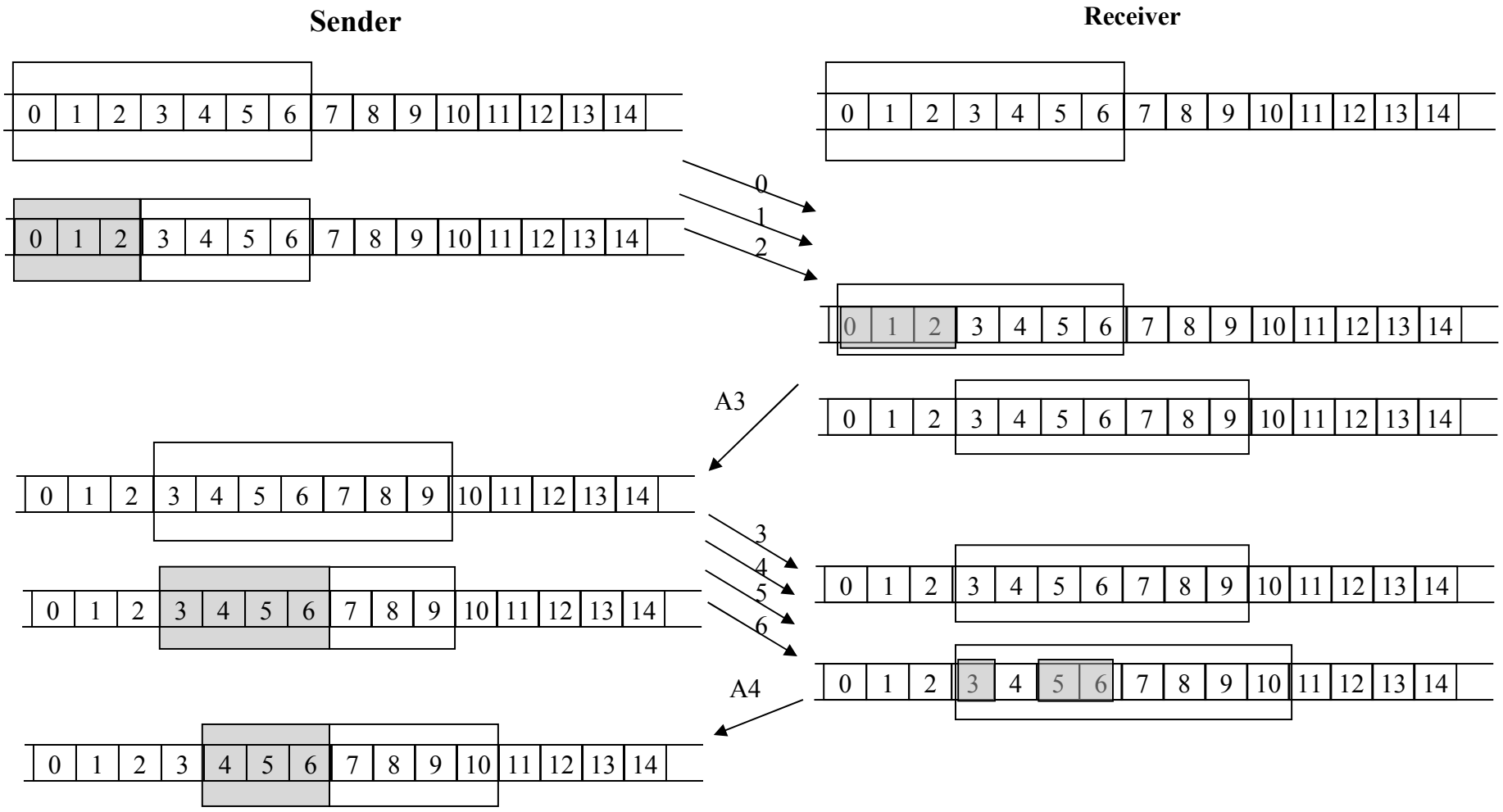
- Maintain three state variables
  - receive window size (**`RWS`**)
  - largest frame acceptable (**`LFA`**)
  - last frame received (**`LFR`**)
- Maintain invariant: **`LFA - LFR <= RWS`**



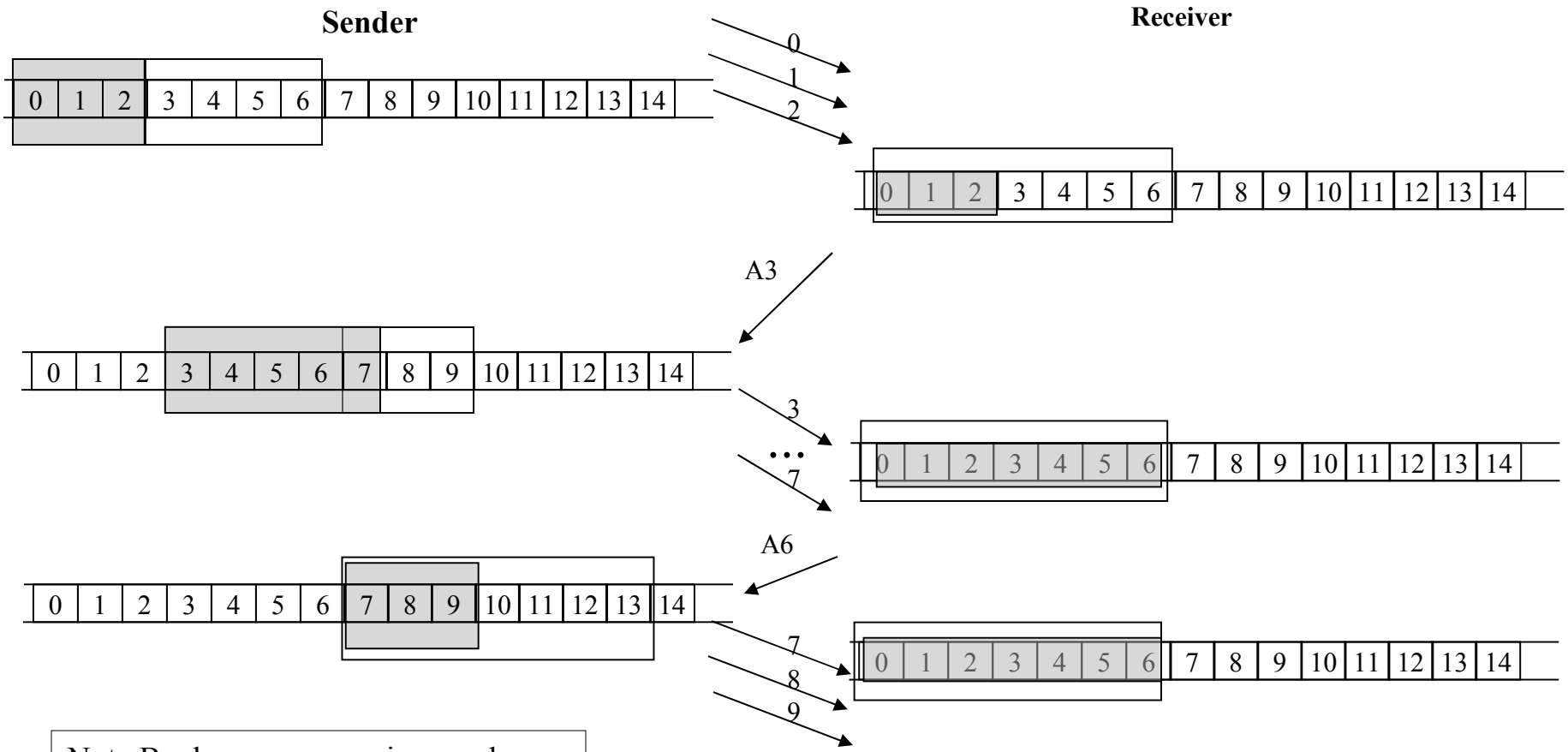
- Frame **`SeqNum`** arrives:
  - if **`LFR < SeqNum ≤ LFA`** ⇒ accept + send ACK
  - if **`SeqNum ≤ LFR`** or **`SeqNum > LFA`** ⇒ discard

- Send *cumulative* ACKs – send ACK for largest frame such that all frames less than this have been received

Sliding Window Example



What If Receiver Client Is Slow?



Note:Book assumes receiver pushes packets up to next higher protocol layer. This slide assumes pull.

No ACKs returned

Sequence Number Space

- **`SeqNum`** field is finite; sequence numbers wrap around
- Sequence number space must be larger then number of outstanding frames
- **`SWS <= MaxSeqNum-1`** is not sufficient
  - suppose 3-bit **`SeqNum`** field (0..7)
  - **`SWS=RWS=7`**
  - sender transmit frames 0..6
  - arrive successfully, but ACKs lost
  - sender retransmits 0..6
  - receiver expecting 7, 0..5, but receives the original incarnation of 0..5
- **`SWS < (MaxSeqNum+1)/2`** is correct rule
- Intuitively, **`SeqNum`** "slides" between two halves of sequence number space

Sliding Window Summary

- Sliding window is best known algorithm in networking
- First role is to enable reliable delivery of packets
  - Timeouts and acknowledgements
- Second role is to enable in order delivery of packets
  - Receiver doesn't pass data up to app until it has packets in order
- Third role is to enable flow control
  - Prevents server from overflowing receiver's buffer