The Data Link Layer

Data Link Layer Design Issues

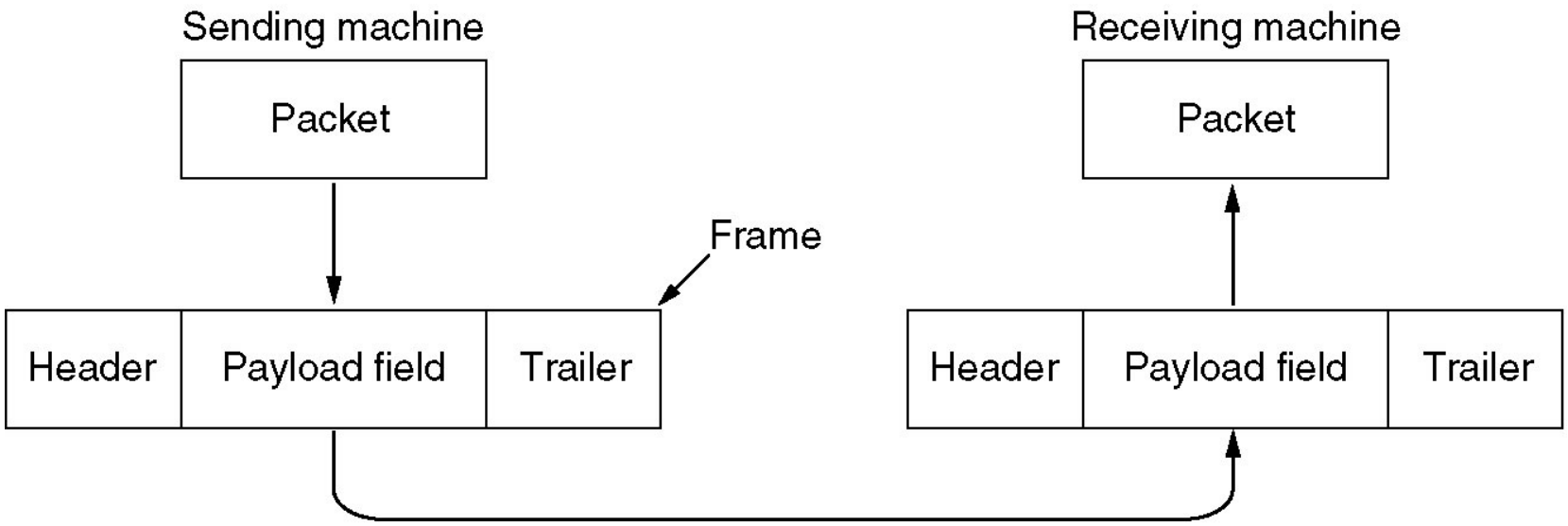
- Services Provided to the Network Layer
- Framing
- Error Control
- Flow Control

Functions of the Data Link Layer

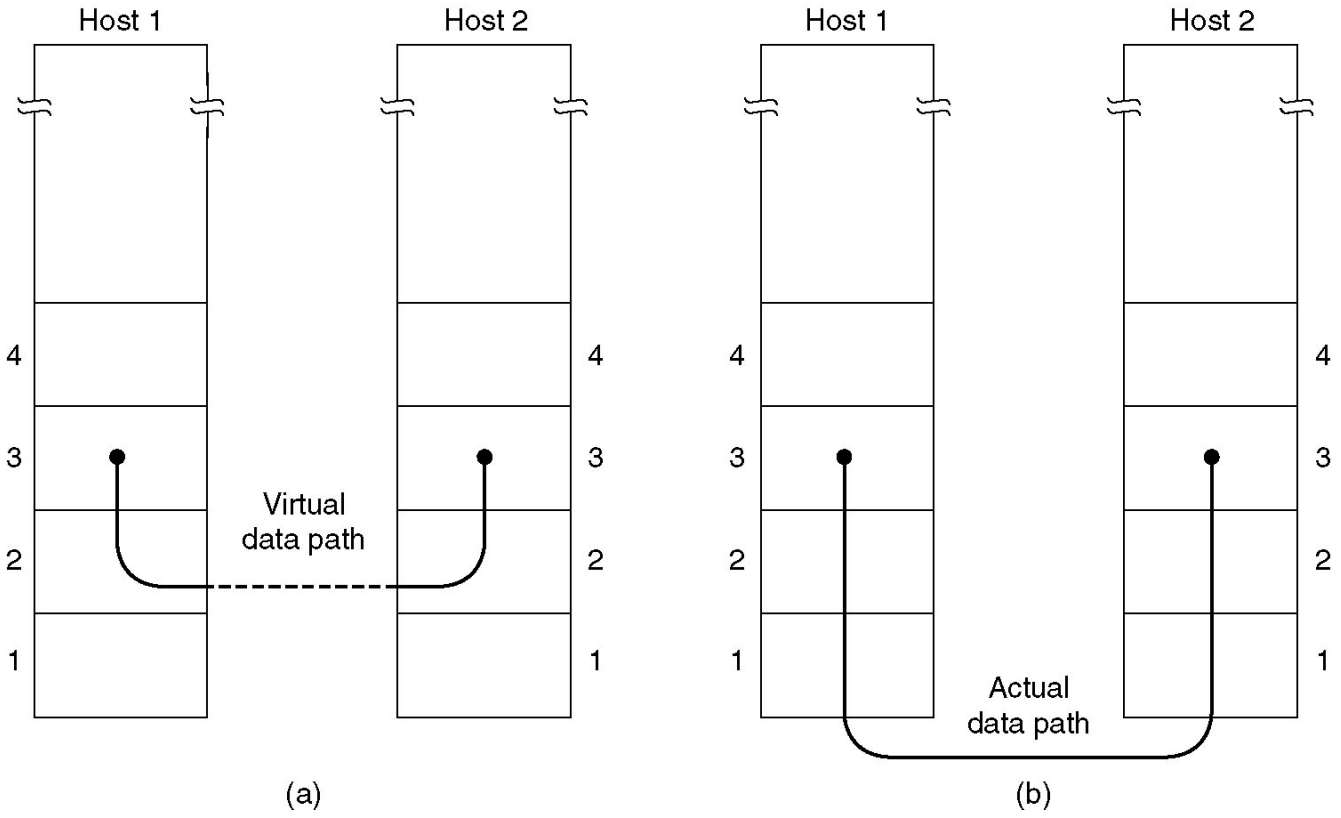
- Provide service interface to the network layer
- Dealing with transmission errors
- Regulating data flow
  - Slow receivers not swamped by fast senders

Functions of the Data Link Layer (2)

Relationship between packets and frames.



Services Provided to Network Layer

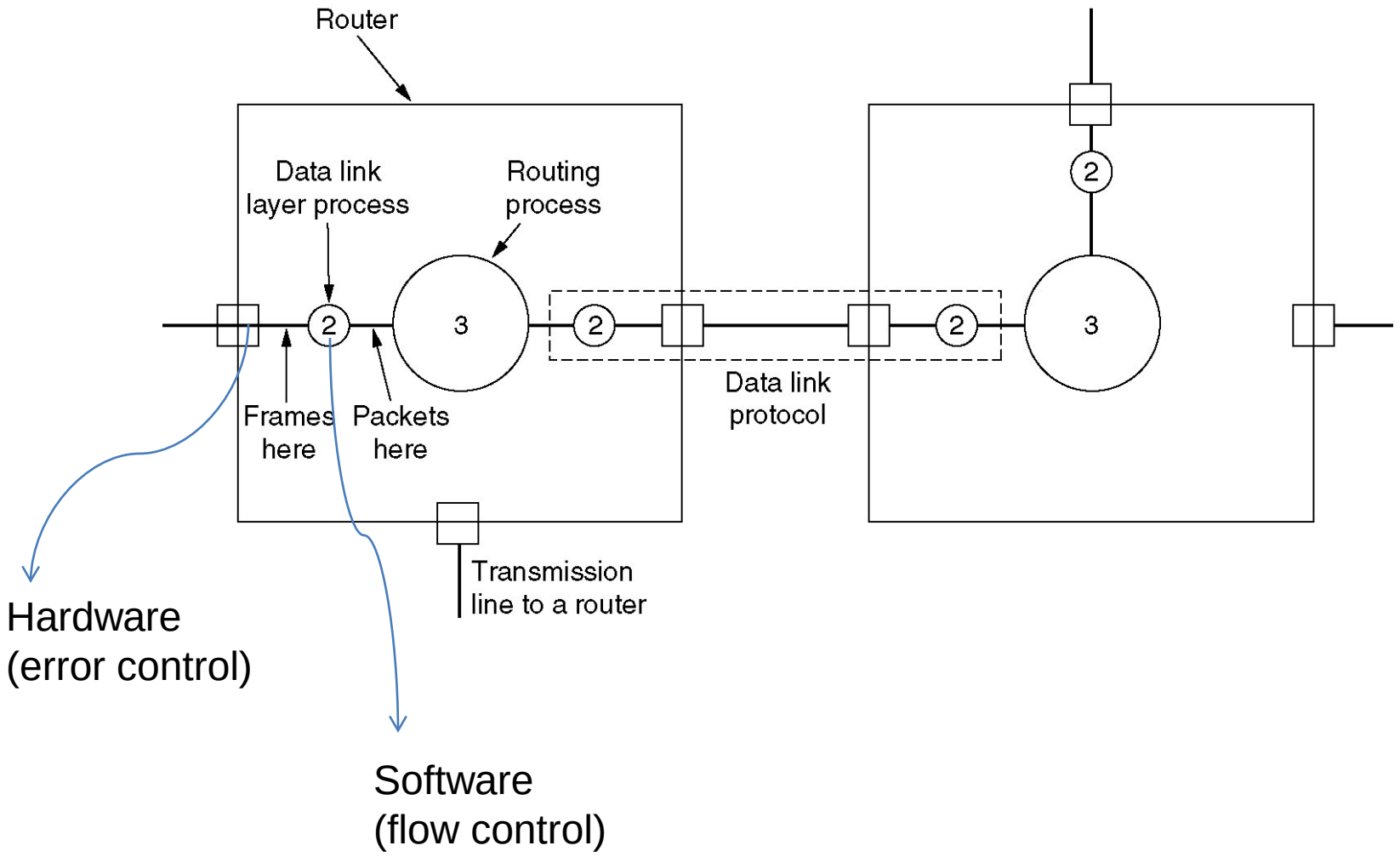


(a) Virtual communication.
(b) Actual communication.

Service types (point to point)

- Unacknowledged connectionless
  - Low error rate, low delay (real time)
- Acknowledged connectionless
  - Useful over unreliable channels such as wireless
- Acknowledged connection oriented
  - Setup, transmit, release

Services Provided to Network Layer (2)

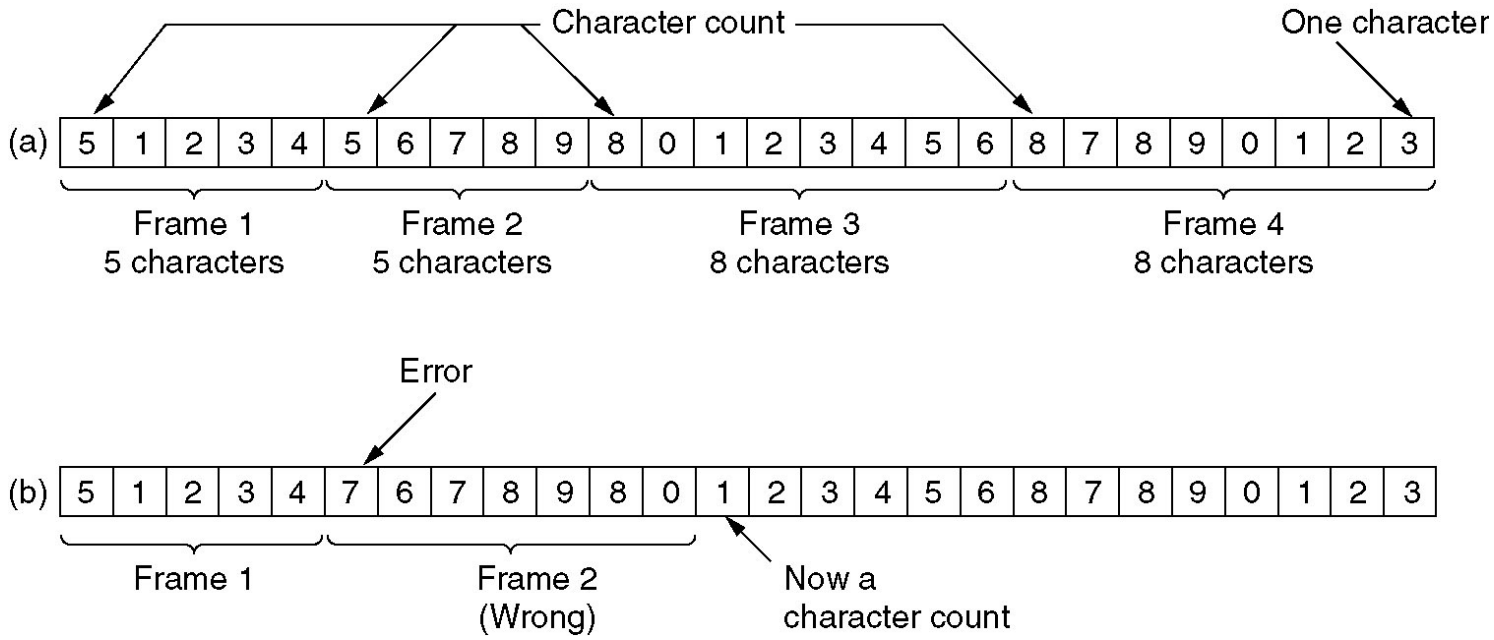


Framing

- Character count
- Flag bytes with byte stuffing
- Starting and ending flags with bit stuffing
- Physical layer coding violations

Framing (Breaking the bitstream up)

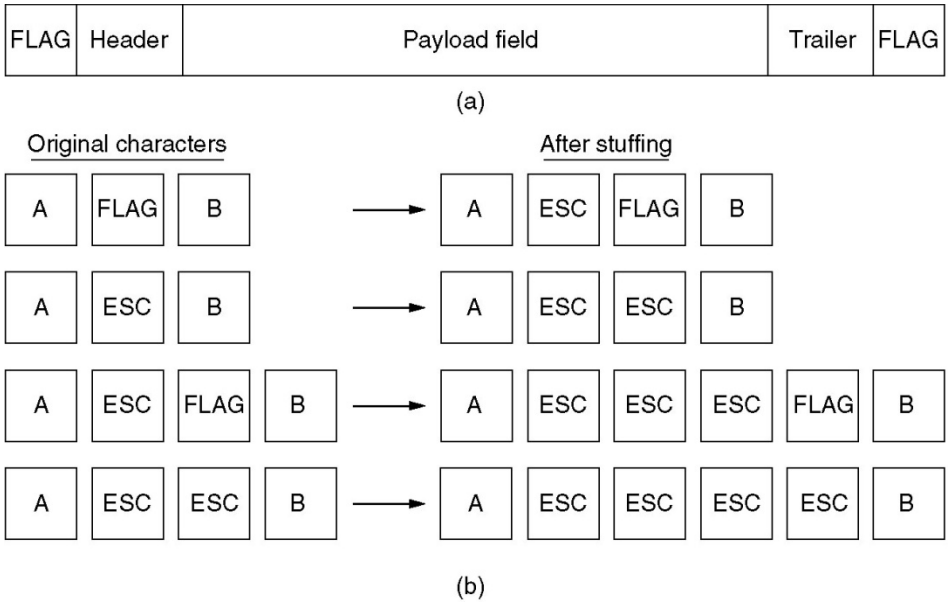
i) Frame delimited by character counts.



i) A character stream (a) Without errors. (b) With one error (synchronization lost!)

Framing (2)

ii) Frame delimited by flag bytes with byte stuffing

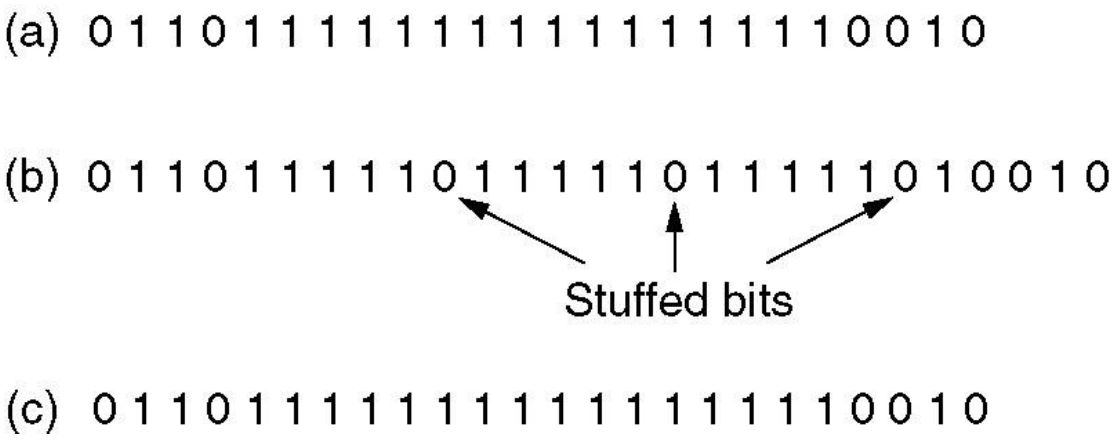


(a) A frame delimited by flag bytes.

(b) Four examples of byte sequences before and after stuffing.

Framing (3)

iii) Frame delimited by flag patterns with bit stuffing (flag pattern 01111110)



Bit stuffing

(a) The original data.

(b) The data as they appear on the line (0 inserted after 11111).

(c) The data as they are stored in receiver's memory after destuffing.

Error Control

- Error-Correcting Codes
  - FEC (Forward Error Correcting)
- Error-Detecting Codes
  - Check for error at the receiver
  - Error: Request a retransmission
  - No error: Either continue or send an ACK

Error-Correcting Codes

- Adding redundancy to overcome errors
- m data (message) bits
- r check bits
- n=m+r codeword bits
- 2^m messages
- 2^n possible codewords
- Code rate: m/n

Hamming distance

$$1010111101 \oplus 0010110101 = 1000001000$$

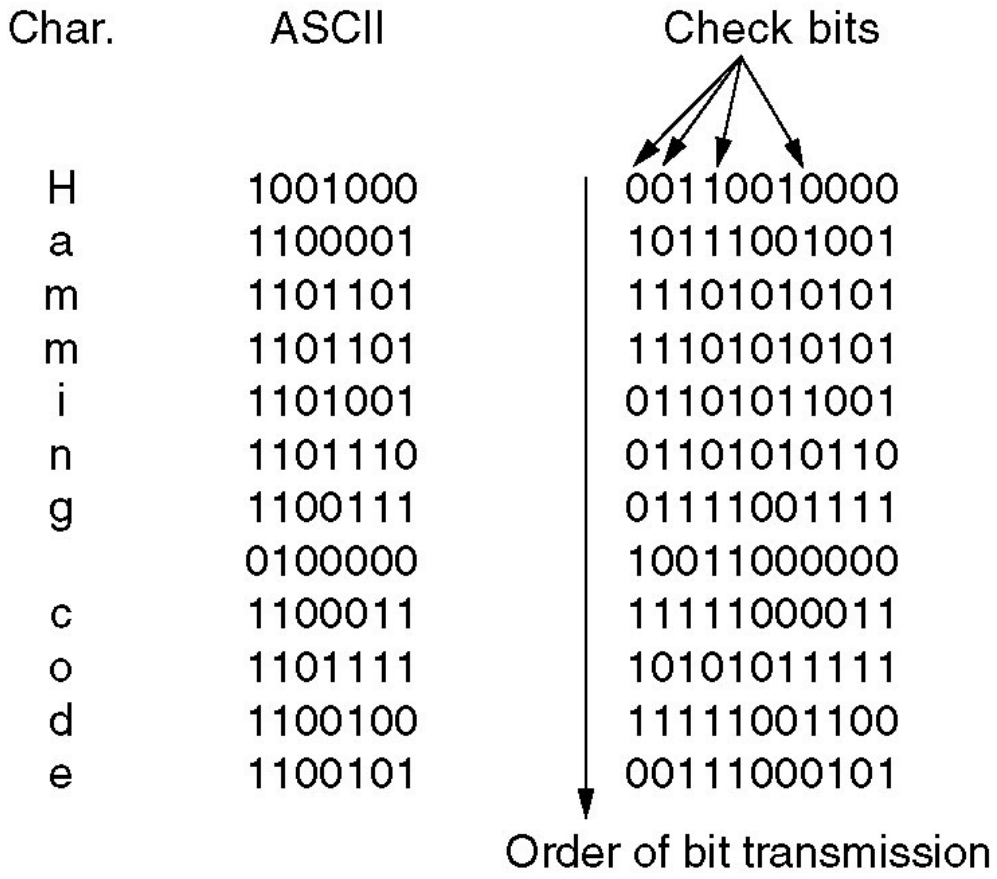
- The two codewords have a Hamming distance of two (differ in two bits)
- Hamming distance of the code: Minimum Hamming distance between any two codewords

Error detection/correction capacity

- Detect d errors: distance d+1 code
- Correct d errors: distance 2d+1 code
- Parity bit:
  - Simple error detecting bit
  - Added so that # of 1 bits is odd/even.
  - d=1

Error-Correcting Codes

Use of a Hamming code to correct burst errors.



Error detecting codes

- Parity codes
  - Odd parity (tek eşlik)
    - 0/1 added to make the sum of 1's equal to an odd number
  - Even parity (çift eşlik)
    - 0/1 added to make the sum of 1's equal to an even number

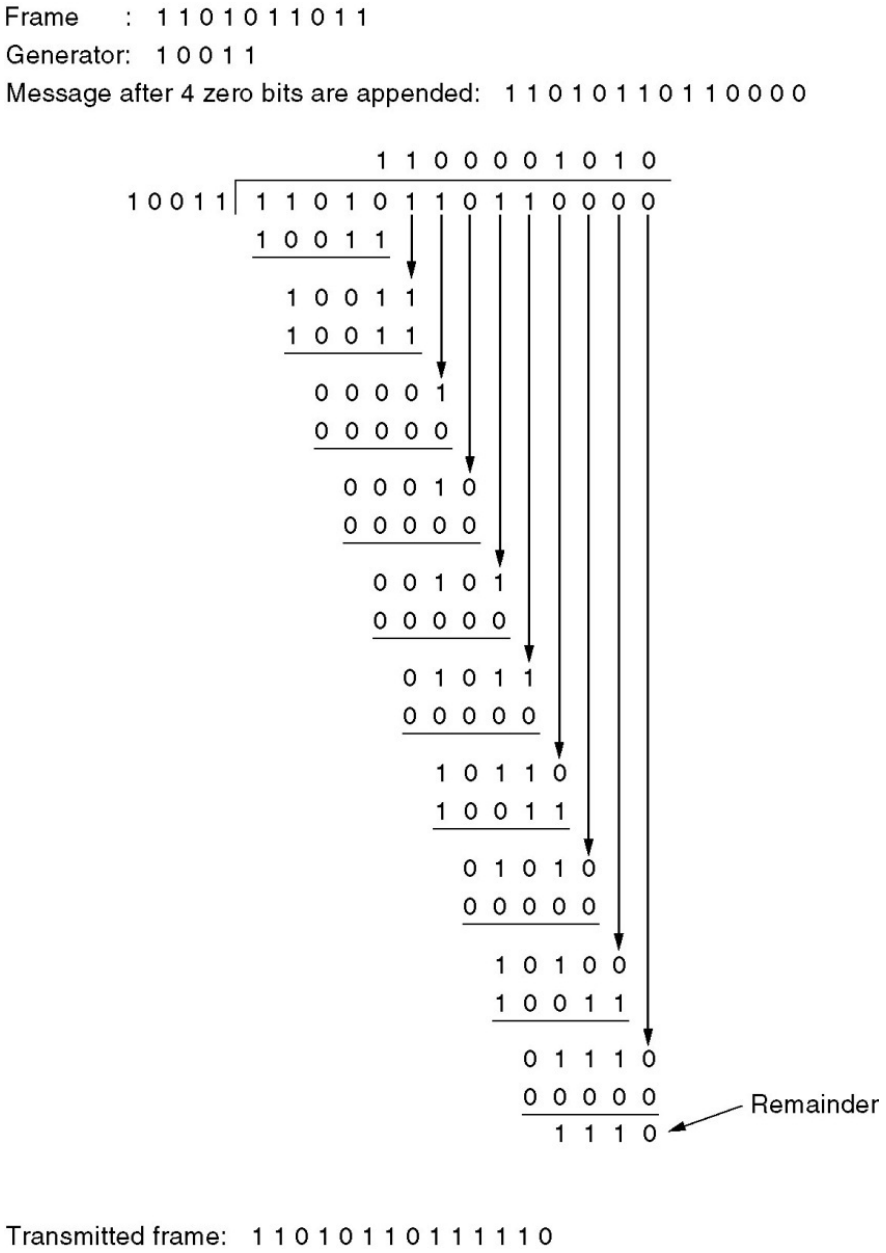
Error detecting codes Cyclic Redundancy Check (CRC)

- Polynomial codes $10100111 \Rightarrow x^7 + x^5 + x^2 + x + 1$
- Polynomial arithmetic is modulo 2.

$$\begin{array}{lll} 10100111 & x^7 + x^5 + x^2 + x + 1 \\ \hline 11010110 & x^7 + x^6 + x^4 + x^2 + x \\ 01110001 & x^6 + x^5 + x^4 + 1 \end{array} \Rightarrow$$

Error-Detecting Codes

Calculation of the polynomial code checksum.



Cyclic Redundancy Check (Çevrimli Fazlalık Sınaması)

- M(x): message polynomial
- G(x): generator polynomial available at transmitter and receiver of degree r $(x^r + \ldots)$
- Transmitter operation:
  - R(x): remainder $R(x) = \dfrac{x^r M(x)}{G(x)}$
  - T(x): transmitted $T(x) = x^r M(x) - R(x)$
- Channel effects $T(x) \xrightarrow{channel} V(x) = T(x) + E(x)$
- Receiver: check $\dfrac{V(x)}{G(x)} \overset{?}{=} 0$

Cyclic Redundancy Check-I (Çevrimli Fazlalık Sınaması)

- Note that by definition $\frac{T(x)}{G(x)} = 0$
- Can catch errors if $\frac{E(x)}{G(x)} \neq 0 \text{ when } E(x) \neq 0$
- Design G(x) to (READ p213 and p214)

$$G(x) = \ldots + x^i + \ldots + x^j + \ldots$$

  - catch single bit errors if it contains two or more terms
  - catch double bit errors:

$$\frac{x^k + 1}{G(x)} \neq 0$$

Cyclic Redundancy Check-II (Çevrimli Fazlalık Sınaması)

– catch odd number of errors, if

$$G(x) = (x+1)\tilde{G}(x)$$

– catch burst errors of length r, if

$$\frac{E(x)}{G(x)} = \frac{x^i(x^{k-1}+\ldots+1)}{(x^r+\ldots+1)} \neq 0 \text{ if } k \leq r$$

Elementary data link layer protocols

- An Unrestricted Simplex Protocol
  - Sender sends frames as long as they come from the network layer
- A Simplex Stop-and-Wait Protocol
  - After sending each frame sender holds until an acknowledgement arrives back from the receiver
- A Simplex Protocol for a Noisy Channel
  - If an ack is not sent by receiver sender times out and retransmits
  - In case the ack gets lost to prevent duplicate frames at the receiver the frame contains a seq. no.

Simplex stop and wait protocol for a noisy channel

- Sender sends frame and starts timer
- Receiver receives frame
  - if checksum is correct sends an ack
  - if seq. no is the expected seq. no. delivers payload to the network layer as a packet
- If sender receives ack in time it increments seq. no. and sends a new frame
- If sender does not receive ack in time it sends a duplicate of the old frame

Sliding Window Protocols

- Simplex to half duplex
  - Piggybacking (sırtında taşıma)
    - Efficient if receiver network layer delivers packets before ack timer expires at sender
- A One-Bit Sliding Window Protocol
- A Protocol Using Go Back N
- A Protocol Using Selective Repeat

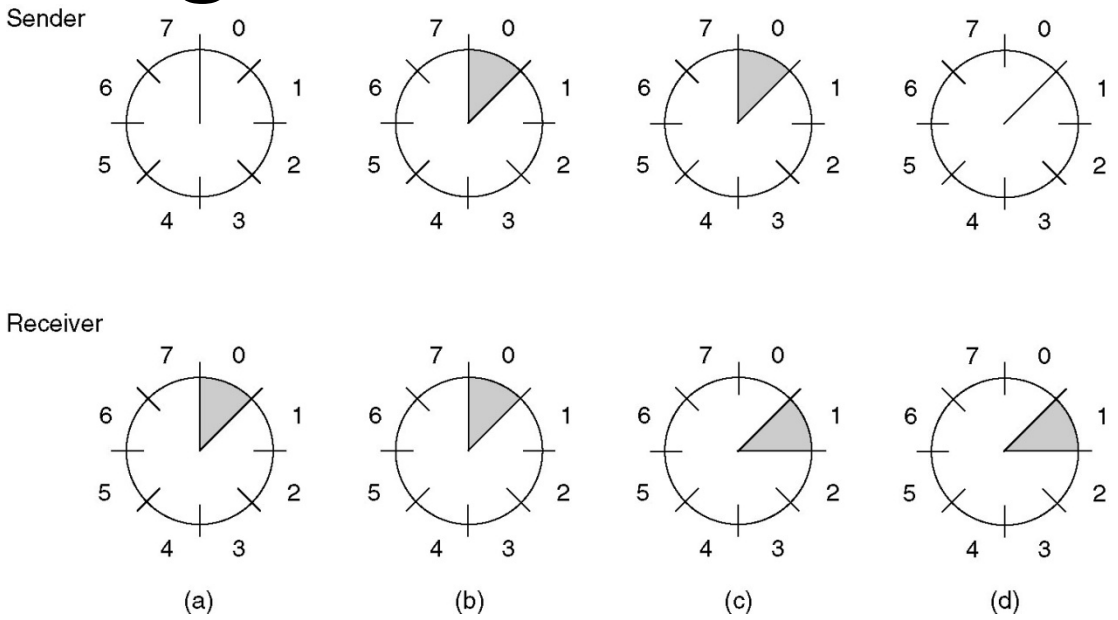
Sliding Window Protocols

- Sequence number fits in an exactly n-bit field ($0 - 2^{n-1}$)
- Sending window, showing the frames permitted to send
- Receiving window, corresponding the frames permitted to accept
- Sending and receiving windows do not have the same bounds or even have the same size.
- In some protocols they have fixed size in some others may shrink or grow in time

Sliding Window Protocols

- If sender window size is n, the sender needs n buffers to keep unacknowledged frames
- Any frame reaching the receiver falling within the window is put in the receiver's buffer.
- Note that a window of size 1 means that the data link layer only accepts the frames in order, but for larger windows it is not so.

Sliding Window Protocols (2)



A sliding window of size 1, with a 3-bit sequence number.

(a) Initially.

(b) After the first frame has been sent.

(c) After the first frame has been received.

(d) After the first acknowledgement has been received.

Sliding Window Protocols

- Pipelining
- Stop&Wait line utilization

$$\frac{l/b}{l/b+R}$$

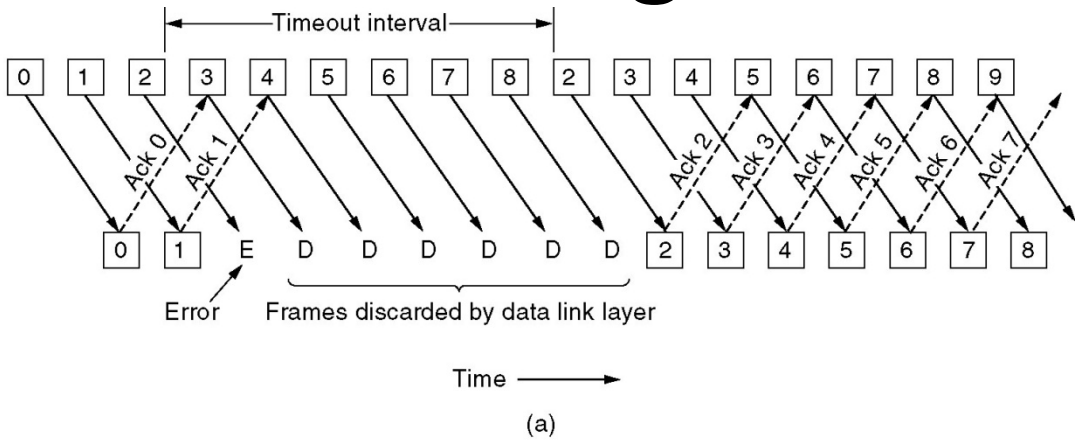
l : *frame size in bits*, R : *round trip delay*, b : *line speed*

- Sending window size : $w = \frac{R + l/b}{l/b}$
- See the example in p232
- Bandwidth-Delay Product of a line!
- Link utilization is important!

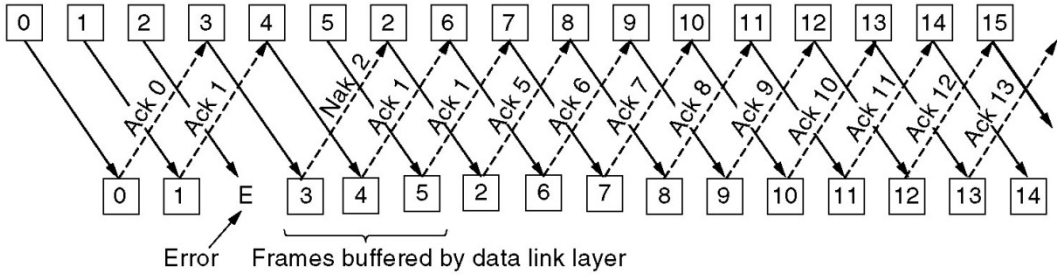
Bandwidth-Delay (BD) product

- BD=bandwidth(bits/sec)*one-way transmit-time
- BD is divided by the number of bits in a frame in order to express it in frames
- In the most efficient system w should be equal to 2BD + 1
- +1 is necessary since an ack frame could not be generated before a complete frame is received.
- Link utilization <= w/(1+2BD)
- Assume that ack frames are rather short.

A Protocol Using Go Back N



(a)



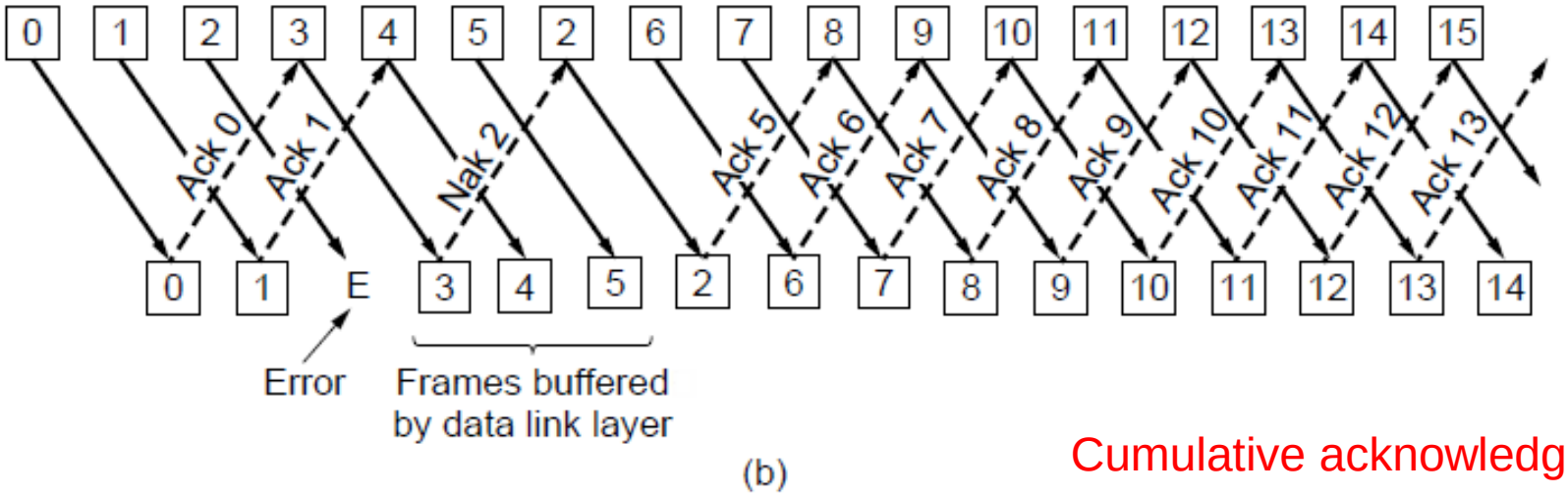
(b)

Pipelining and error recovery. Effect on an error when

(a) Receiver's window size is 1.

(b) Receiver's window size is large (selective repeat).

Protocol Using Go-Back-N (2)



(b)

Cumulative acknowledgement!

Pipelining and error recovery. Effect of an error when

(b) receiver's window size is large.

Go-Back N → Selective Repeat

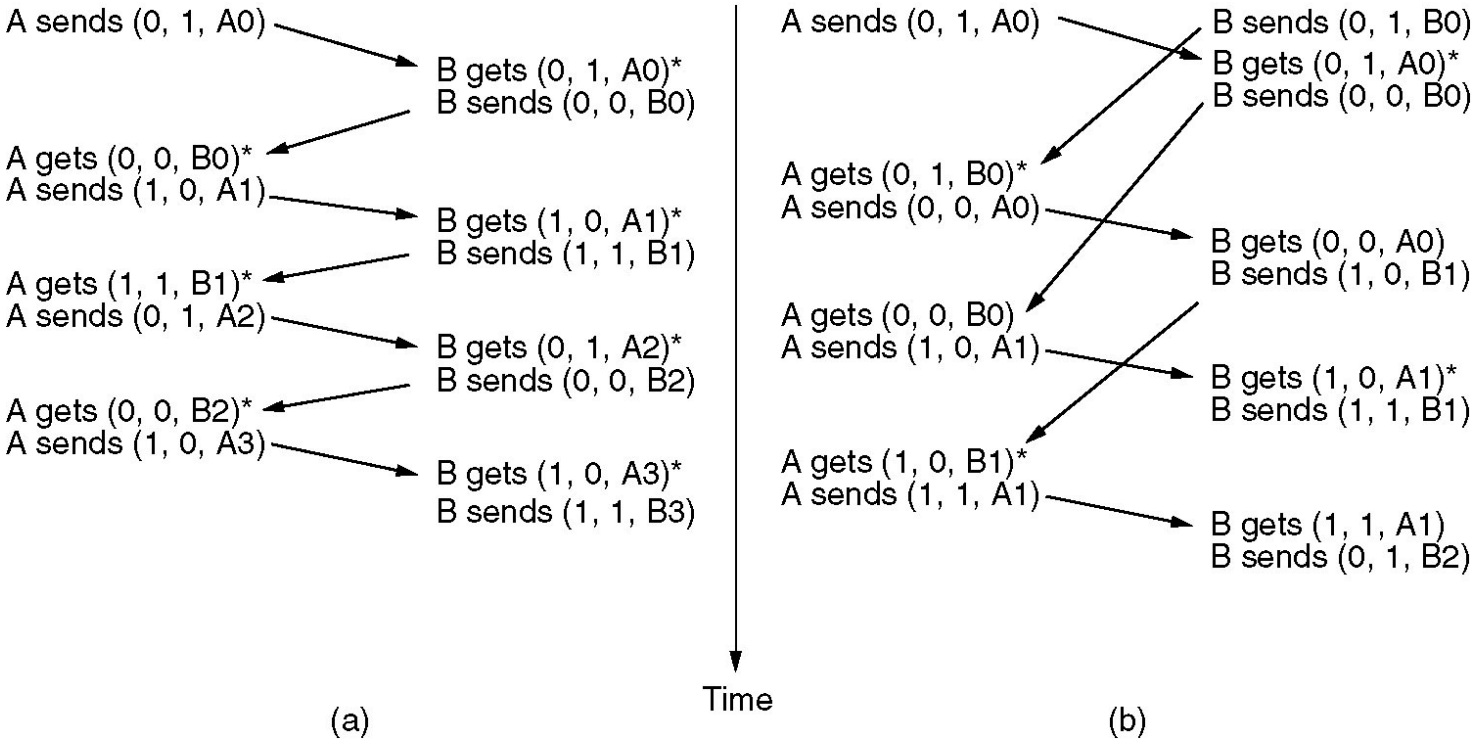
- Window size should be MAX_SEQ – 1

Why?

- The sender sends frames from 0 to 7 (when max_seq=7)
- A piggybacked ack for 7 comes to sender
- The sender sends another 8 frames 0 – 7
- Another piggybacked ack for frame 7 comes

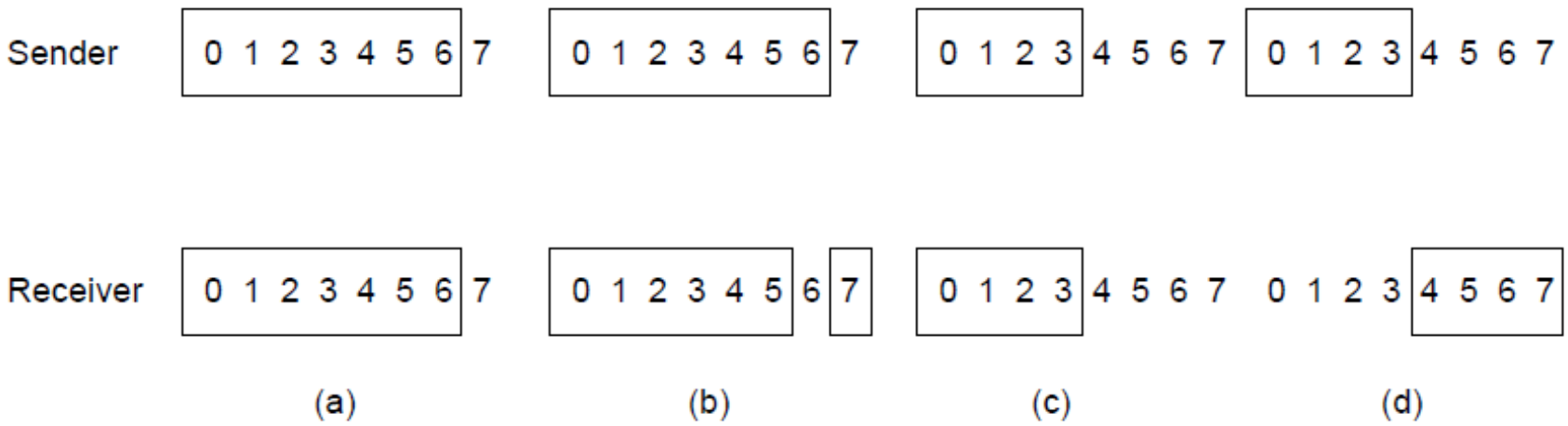
* When the frames of the second batch are all lost the receiver will be sending frame 7 as the ack.

A One-Bit Sliding Window Protocol (2)



Two scenarios for protocol 4. (a) Normal case. (b) Abnormal case. The notation is (seq, ack, packet number). An asterisk indicates where a network layer accepts a packet.

A Protocol Using Selective Repeat



a) Initial situation with a window of size7

b)After 7 frames sent and received but not acknowledged.

c) Initial situation with a window size of 4.

d) After 4 frames sent and received but not acknowledged.

Window size = (MAX_SEQ +1)/2

End of Chapter 3