# Lecture 4
# Data Link Layer

Dr. Thaier Hayajneh
Department of Computer Engineering
The Hashemite University

408450 Computer Networks, Fall 2011/2012
http://www.hlms.hu.edu/

## Topics

- Introduction
- Flow control
  - Stop and wait
  - Sliding window
- Error detection
  - Parity
  - Checksums
  - Cyclic redundancy check (CRC)
- Error handling
  - Error correction
  - Retransmission (*Automatic Repeat Request* ARQ)
    - Stop and wait
    - Go-back-N
    - Selective reject (selective repeat)



## Data Link Layer: Background

- Physical layer provides means to transfer *frames* over a link:
  - Physical medium
  - Data transmission with electromagnetic waves
    - Line coding (low-pass channel)
    - Modulation (band-pass channel)
  - Synchronization
- Remaining problems to be solved
  - Adapt sender to receiver rate
  - Errors in frames and lossage of frames should be detected and managed
  - ...

| 8-bit flag | Header— address and control | Data | Trailer— error control | 8-bit flag |
|---|---|---|---|---|

## Flow Control: Assumptions and Problems

**Initial**

- **Simplex Protocol**
- **Infinite buffer capacity with the receiver**
- **Error free transmission**
- **Network layer at the senders end is always ready with data**
- **No need for flow control**

**Finite capacity in the buffer of the receiver.**

- **Need for "flow control"**
- **Stop-n-Wait protocol**
  - **Sender sends a frame and waits for a signal in the form of a dummy frame**
  - **No seq no. is required since the line is still error free**

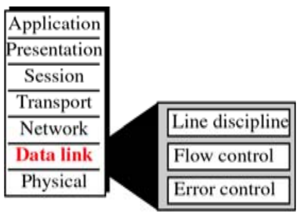
**The channel is noisy, frames may be damaged or lost**

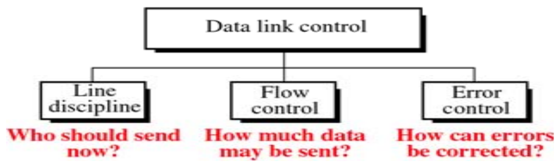
- **Good scene : data frame reaches intact, ack sent back and received, next frame sent**
- **Bad scene :**
  - **Data frame damaged or lost hence no ack – sender times out and resends .. No problems**
  - **Data frame reaches intact but Ack lost ..Times out ..resends.. Receiver receives duplicate frames. Problem!!**

**Solution :**
**Keep a sequence number for each frame to distinguish between the new frame and a duplicate frame.**

## Data Link Layer: Background

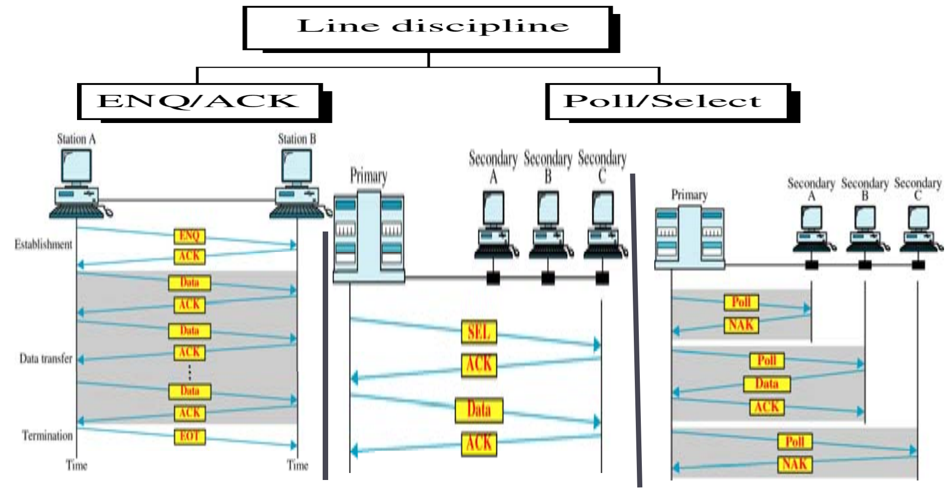
- Data link layer is responsible for hop-to-hop packet delivery (local responsibility).
- Flow Control and error control are the main functions of the data link layer.






## Line Discipline






## Line Discipline

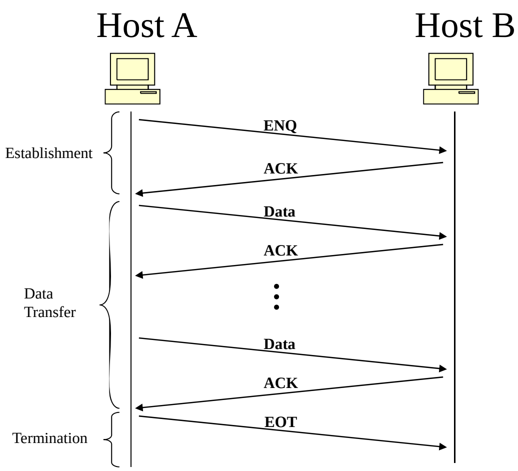
- Determine the direction of communication
- Make sure that receiver is ready to accept or signal the sender to start
- Two ways:
  - Enquiry / Acknowledgment (ENQ/ACK)
    Dedicated line between hosts
  - Poll / Select
    Multipoint connections



## ENQ/ACK

- Dedicated line between hosts, no problem of addressing
- Coordinates which device may start transmission, and if the receiver is ready and enabled
- If both hosts have equal ranks, either can initiate the process
- Otherwise, only higher-ranked host is allowed to start the transmission request
- Can be run in either half-duplex or full-duplex modes

## ENQ/ACK




- Establishment: Host B responds either with ACK or NAK
- Host A tries to send ENQ three times before concluding that Host B is down

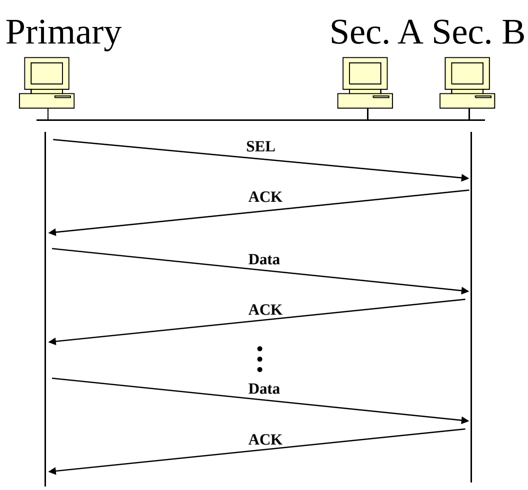


## Poll / Select

- Multipoint connections
- One primary and multiple secondary hosts
- Communication between secondary devices go over the primary
- **Select** mode is used when primary has something to send to a secondary (downstream)
- **Poll** mode is used to solicit transmissions from a secondary to the primary (upstream)
- Address must be contained in all packets



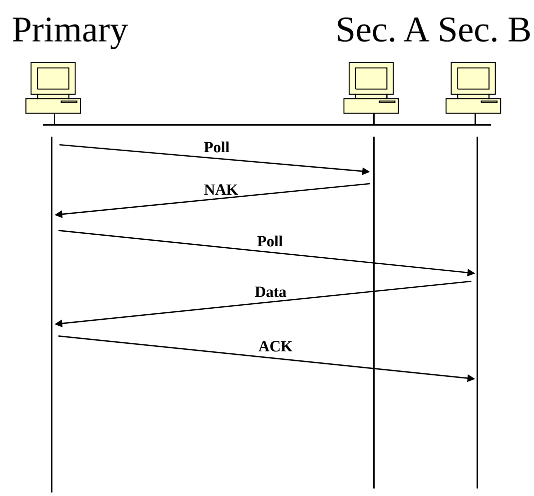
## Poll / Select




- Select mode
- SEL packet contains address of B
- B can response either by ACK or NAK
- Primary sends one or more data packets, which are ACKed by B



## Poll / Select




- Poll mode
- Poll packet contains address of recipient
- If the intended secondary has no data to send, replies with NAK
- Data is ACKed by the primary

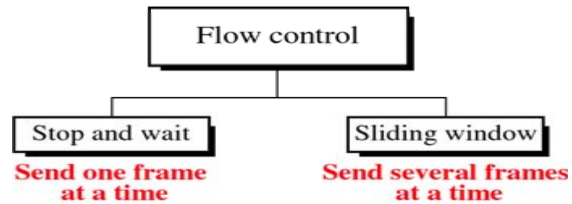
# Why Flow Control?

- Problem: Sender can overload receiver
  - Frames arrive too fast
    - In many cases, the receiver is more complicated than the sender
    - Error detection, frame/packet analysis, address lookup
  - Frames are stored in a buffer before they are processed
    - Receiver buffers can overflow and frames be lost
  - Prevent loss of frames
- Combined mechanisms for flow control and error control
  - Based on retransmission



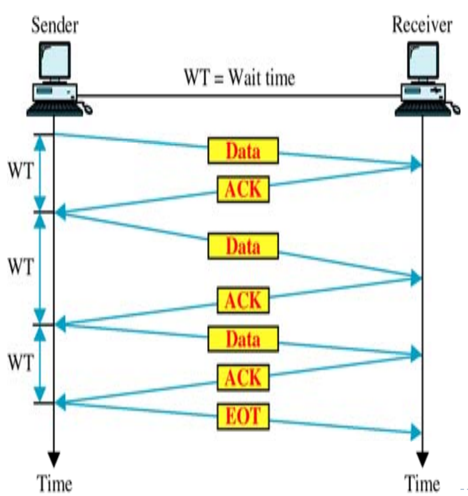
# Flow Control

- Control mechanisms
  - Stop and wait
  - Sliding window
- (We don't worry about frame errors and losses for now)





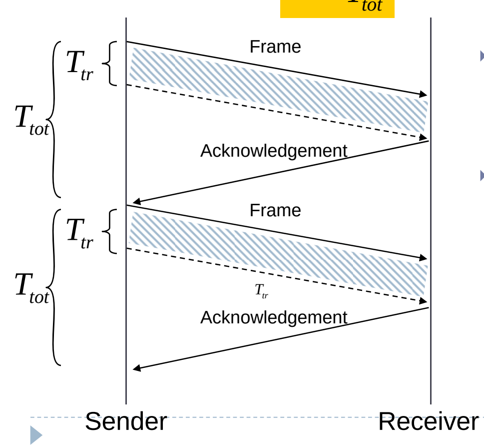

# Stop and Wait




- Sender sends one frame
  - Waits for acknowledgement
  - Next frame sent after acknowledgement
- Receiver sends acknowledgement
  - When ready to receive next frame

# Link Utilization

$$U = \frac{T_{tr}}{T_{tot}}$$



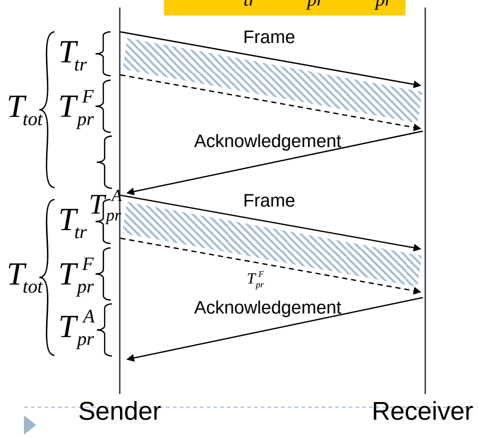

- Transmission time
  - Time between first and last bit of a frame
    - Frame length (bits) divided by link capacity
- Total time
  - Time from first bit is sent until acknowledgement arrives
    - How long before sender can start sending again
- Assumptions
  - Zero transmission time for acknowledgements
  - Zero processing time in sender and receiver

$T_{tr}$ Transmission time

$T_{tot}$ Total time

## Link Utilization

$$U = \frac{T_{tr}}{T_{tr} + T_{pr}^{F} + T_{pr}^{A}}$$




- Propagation time
  - Time for one bit to propagate over the link
    - Link length divided by propagation speed

$T_{pr}^{F}$ Propagation time for frame

$T_{pr}^{A}$ Propagation time for acknowledgement

## Utilization

- How large part of the total time ($T_{tot}$) is used for transmission?
  - $T_{tr}$ is the time to send a frame (transmission time)
    - Time between first and last bit of the frame
    - Given by the frame length (bits) divided by the link capacity (b/s)
  - $T_{pr}$ is the propagation time of the link
    - Time to distribute a bit over the link
    - Given by the link length divided by the signal propagation speed
      - Approximately speed of light
  - Link is characterized by the parameter $a = T_{pr} / T_{tr}$

$$U = T_{tr} / T_{tot} = T_{tr} / (T_{tr} + 2T_{pr}) = 1 / (1 + 2a)$$

  - Large $a$ means poor link utilization



## Link Utilization—Symmetrical Links

For symmetrical links:

$$T_{pr}^{F} = T_{pr}^{A} = T_{pr}$$

$$U = \frac{T_{tr}}{T_{tr} + 2T_{pr}} = \frac{1}{1+2a}, \quad \text{where } a = \frac{T_{pr}}{T_{tr}}$$

- The parameter $a$ is the relation between length of link and "length" of frame (in meters)
- "Length" of a bit:
  - Link capacity divided by signal propagation speed
  - Speed of light in optical fiber is about $2 \times 10^8$ m/s
    - 1 kb/s: 200 km
    - 1 Mb/s: 200 m
    - 1 Gb/s: 20 cm

## Utilization

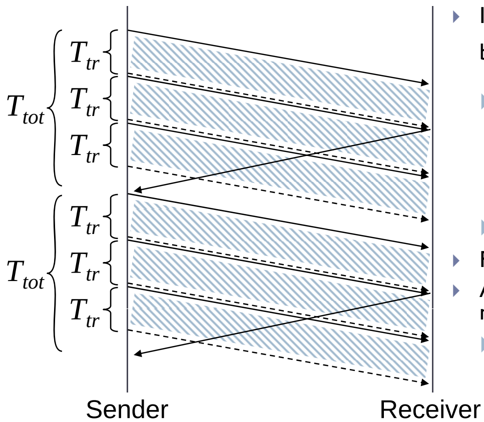
$T_{pr} < T_{tr}$ $(a < 1)$: max one frame fits on the link

| a | U | Situation |
|---|---|---|
| **0,01 to 0,1** | 0,98 to 0,83 | LANs |
| **$10^{-5}$** | 0,99998 | Modem, 100m |
| **0,48** | 0,51 | Modem, 5000km |

$T_{pr} > T_{tr}$ $(a > 1)$: multiple frames on the link

| a | U | Situation |
|---|---|---|
| **3,8** | 0,12 | 4kb frame on 56kbps satellite link |
| **2160** | 0,000231 | 4kb frame on 32M bps satellite link |

## Sliding Window




- Increase utilization by sending *N* frames before waiting
  - Need flow control mechanism
    - To prevent buffer overflow at the receiver
  - *N* is "window size"
- Frame sequence numbers
- Acknowledgements contain sequence numbers
  - Sequence number of the next frame the receiver is ready to accept

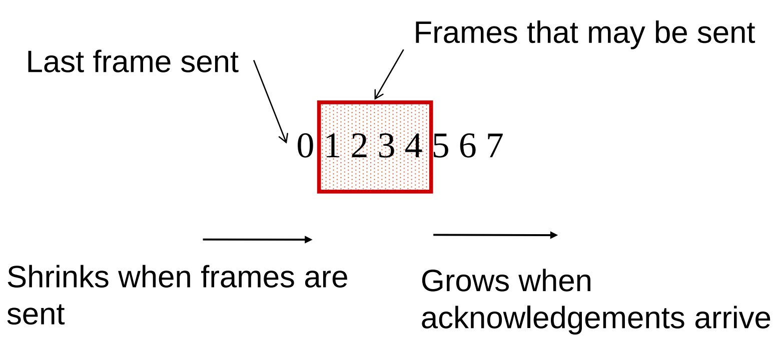
## Sliding Window

- Frame are numbered
  - Sequence number
- The sender may send *N* frames before receiving an acknowledgment
  - *N* is the window size
- The receiver acknowledges frames by sending the sequence number of the next expected frame
  - An acknowledgement means that the receiver is prepared to receive *N* more frames, starting from the sequence number specified in the acknowledgement
  - Optimization: acknowledge multiple frames with the same acknowledgement



## How Does it Work?

### At the sender (*N* = 4)



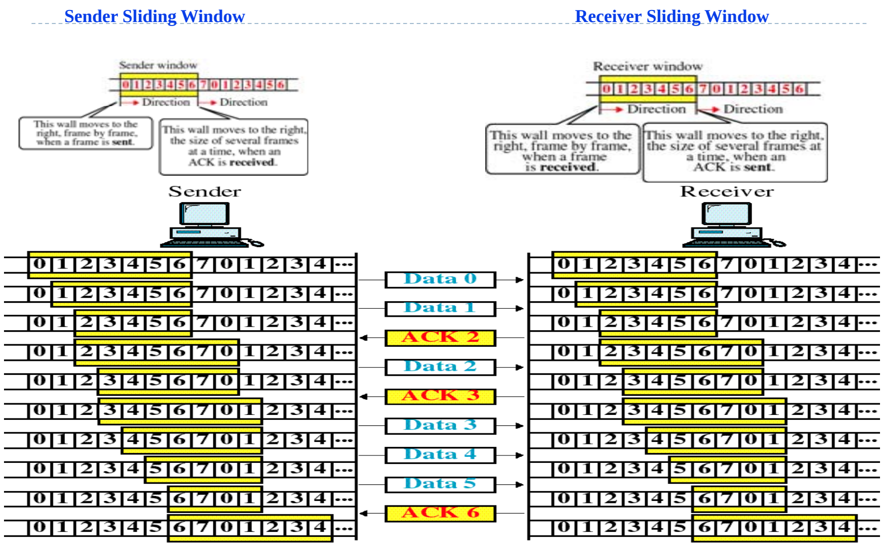



### At the reciever

Receiver window:

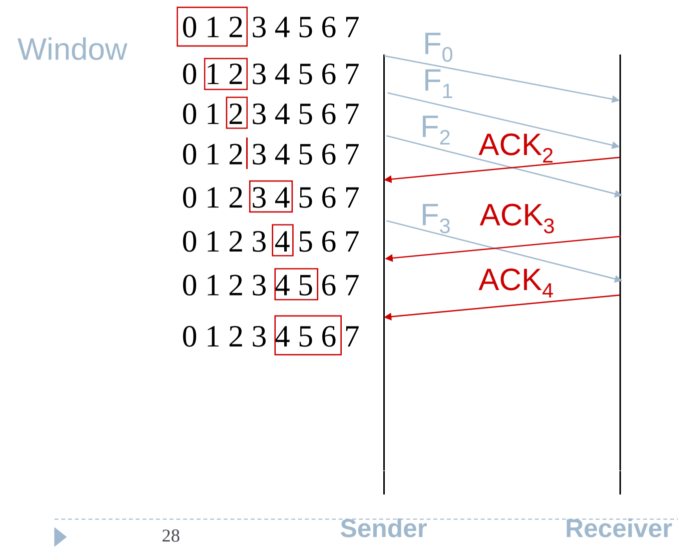
Shrink from left as frames are received

Expand from right as ACKs are sent



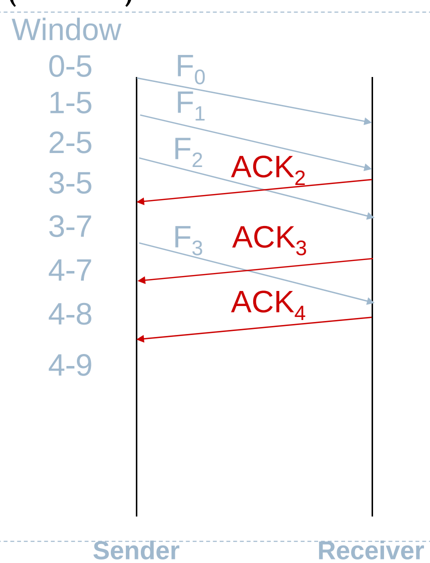



# Example (N = 3)






# Example (N = 6)






# Utilization

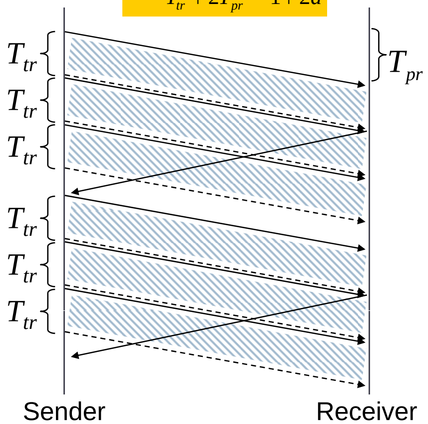
$$U = N \times T_{tr} / T_{tot} = N \times T_{tr} / (T_{tr} + 2T_{pr}) = N / (1 + 2a)$$

where $a = T_{pr} / T_{tr}$

- $N \times T_{tr} > T_{tr} + 2T_{pr} \Rightarrow U > 1$
  - Sender receives acknowledgement before window is closed
  - Sender may send without stopping
  - (Although "true" utilization can never be more than 100%)
- $N \times T_{tr} < T_{tr} + 2T_{pr} \Rightarrow U < 1$
  - Window closes after $N \times T_{tr}$
  - Sender must stop and wait for acknowledgement
  - Utilization is the fraction of the time when the sender does not wait

# Sliding Window Utilization

$$U = \frac{NT_{tr}}{T_{tr} + 2T_{pr}} = \frac{N}{1+2a}$$




- $U > 1$
  - Sender receives acknowledgement before window is closed
  - Sender may send without stopping
  - (Although "true" utilization can never be more than 100%)
- $U < 1$
  - Window closes after $N$ frames
  - Sender must stop and wait for acknowledgement
  - Utilization is the fraction of the time when the sender does not wait

# How Large Window?

- $N$ = 1 ⇒ stop-and-wait
- Small $a$ ⇒ small $N$
  - Local area network: $N$ = 8 ⇒ 3 bits
- Large $a$ ⇒ large $N$
  - TCP uses 32-bit sequence number
    - Byte number



# Acknowledgements

- Types of acknowledgements
  - Positive
    - ACK (*acknowledgement*)
      - HDLC: RR (receiver ready)
  - Negative
    - NACK (*negative acknowledgement*)
      - HDLC: RNR (receiver not ready)
- Indicates sequence number of next expected frame
- When and how is the acknowledgement sent?
  - As a separate frame
  - Together with data from the receiver to the sender
    - "Piggybacking"



# Sliding Window Protocols

- A One-Bit Sliding Window Protocol
- A Protocol Using Go Back N
- A Protocol Using Selective Repeat

# Automatic Repeat Request (ARQ)

- Error control—when frames or acknowledgements are lost
  - Based on flow control
- Stop-and-wait flow control
  - Stop-and-wait ARQ
  - “Alternating Bit Protocol”
    - Two sequence numbers—0 and 1
- Sliding window flow control
  - Go-back-N ARQ
  - Selective-reject ARQ



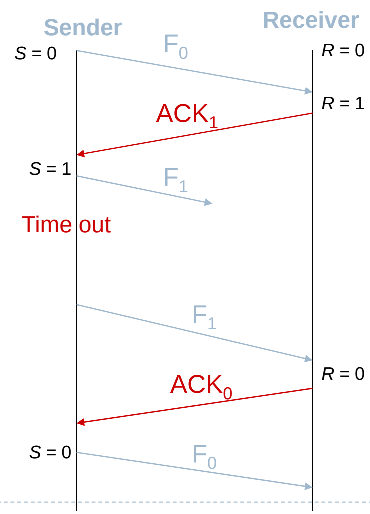
# Stop-and-wait ARQ

- Positive acknowledgements ACK
- Problem: acknowledgements can be lost or delayed
  - Therefore the acknowledgements are numbered
  - Indicates the sequence number of the next expected frame
- Alternating Bit Protocol—sequence numbers 0 and 1



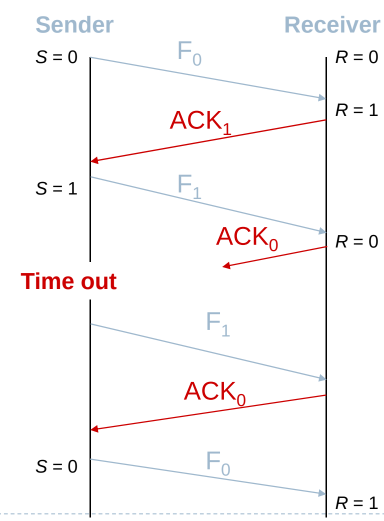
# Stop-and-wait ARQ

- Sender
  - Variable *S*: sequence number of last frame sent
  - Keeps a copy of last frame sent
  - Starts a timer when a frame is sent
  - Stops timer when ACK is received
  - Retransmits if time out (and restarts timer)
- Receiver
  - Variable *R*: next expected sequence number
  - When a frame is received, sends an ACK with next expected sequence number
  - Drops received packet if wrong sequence number




# Stop-and-wait ARQ: Lost Acknowledgement

- No ACK for $F_1$
- Sender time out
  - Retransmission
- Receiver receives wrong sequence number
  - Discards frame
  - Sends ACK with expected sequence number (0)
- Sender may send next frame



## Continuous ARQ

- Stop-and-wait ARQ is simple but inefficient
- Continuous ARQ (multiframe ARQ)
  - Sequence numbers with sliding window
  - ACK and NACK
  - Time out

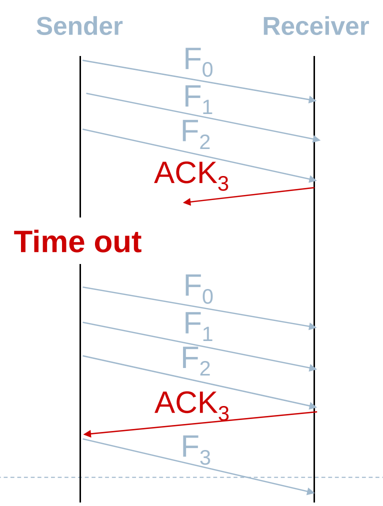


## Go-back-N ARQ

- Based on sliding window flow control
- Sender
  - May send *N* frames without acknowledgement
    - Copies of all unacknowledged frames are kept in a buffer
  - Time out:
    - retransmit all unacknowledged frames
- Receiver
  - Discards frames with unexpected sequence numbers



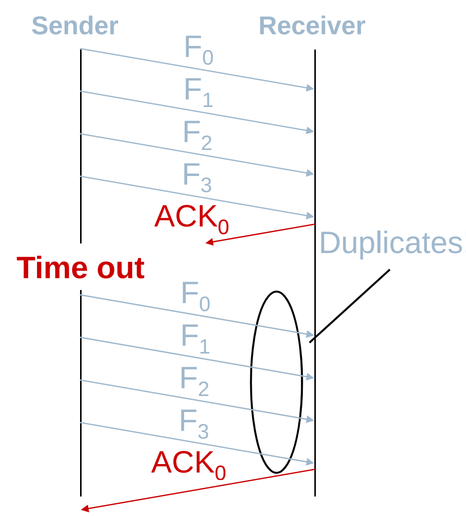
## Example: Lost Acknowledgement ($N = 3$)






## Window Size Versus Sequence Numbers

- With *k*-bit sequence numbers, window size can be at most $2^k$-1
- For example:
  - Sequence numbers 0-3 ($k = 2$)
  - Window size $2^k = 4$ (incorrectly)

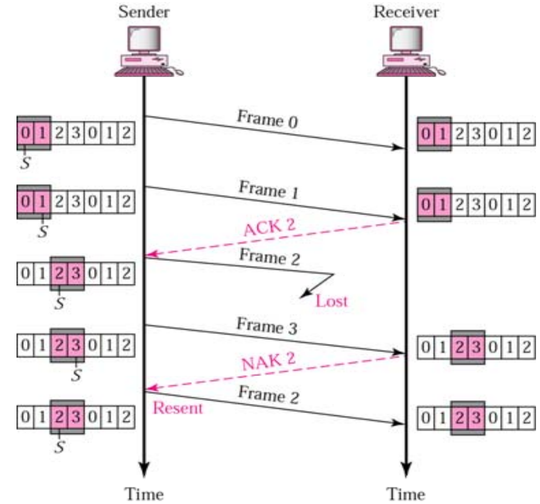


# Selective Repeat ARQ

- Sometimes also called Selective Reject ARQ (SREJ)
- Only retransmit frames that are lost
  - Negativ acknowledgement NAK (SREJ)
  - Time out
- Receiver has a receive window
  - Only frames with sequence number within receive window are accepted
- Advantage
  - Minimizes the number of retransmissions
  - More suitable for noisy links
- Disadvantages
  - More buffering at receiver
    - Needs to keep out-of-order frames in a buffer
- Window size cannot be larger than one-half the number of sequence numbers

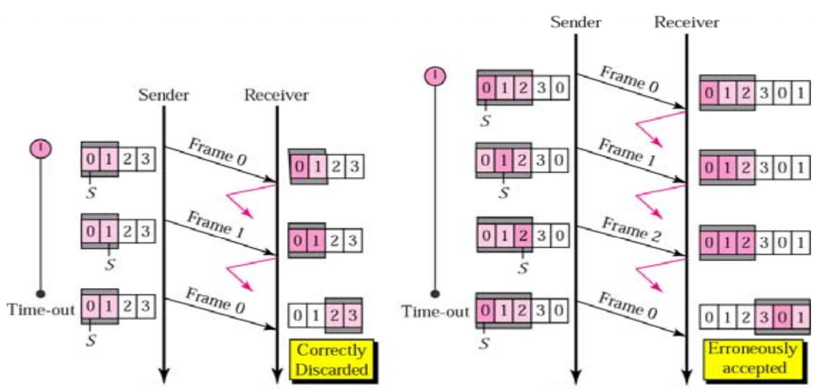


# Selective Repeat ARQ






# Window Size in Selective Repeat ARQ






# Transmission Errors

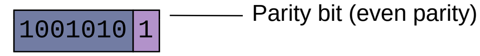
- Lost frame
  - Framing error
- Corrupted frame (bit errors)
- Single bit error
- Burst errors
  - Whole sequences of bits are corrupted
  - External noise

# Error Detection—Basic Idea


- Add extra (redundant) information for detecting errors
  - Parity check
  - Checksum
  - Cyclic redundancy check (CRC)
- Sender computes function over data, and appends result
- Receiver computes same function, and compares the results
- If the results differ, there was an error



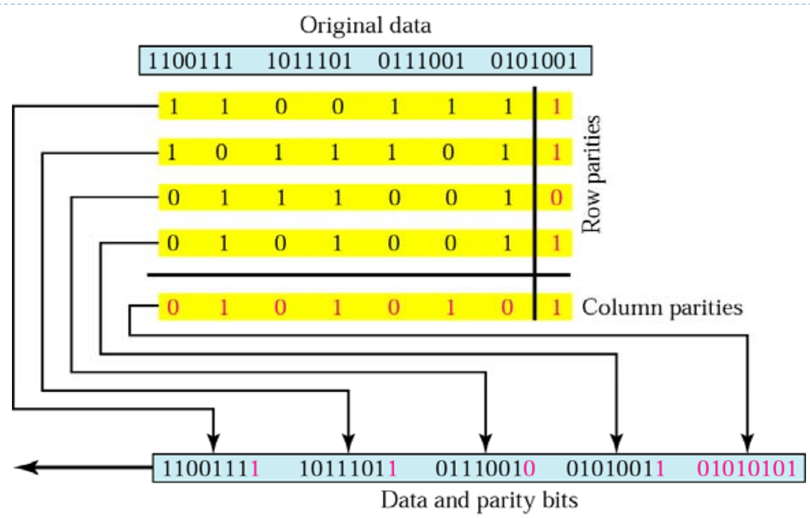
# Parity Check




- Simple parity check: extra bit (parity bit) is added to the data unit
  - Numbers of 1s in the unit is always even ("even parity") or odd ("odd parity")
  - Receiver checks number of 1s
- Advantages
  - Simple: $P = 1 \oplus 0 \oplus 0 \oplus 1 \oplus \ldots \oplus 1 \oplus 0$ for even parity
  - Inexpensive: cost is one extra bit per data unit
- Disadvantage
  - Only detects single bit errors, and burst errors with odd number of bit errors

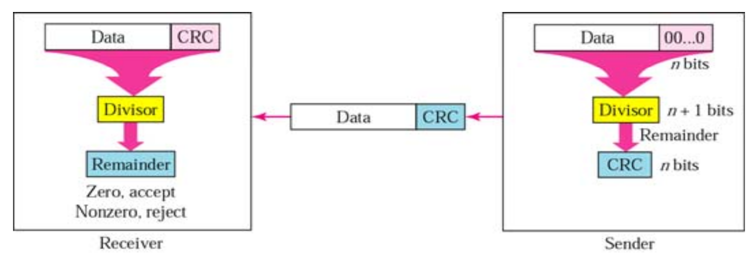


# Two-dimensional Parity






# Cyclic Redundancy Check (CRC)




- The data *M* is treated as a sequence of bits
- Predefined binary word *P* (generator) of length *n*+1
- Sender generates *M*' by adding *n* CRC bits to *M*
  - Such that *M*' is evenly divided by *P*
  - *M*' is sent
- Receiver receives *M*''
  - If remainder of *M*'' divided by *P* is zero then *M*'' = *M*'
  - Otherwise: bit error detected, discard the data

## CRC Calculation Using Binary Division

- Append '00' to *M*
- Binary subtraction (xor) of 3 bits
  - If first bit is '1'
    - subtract *P*
    - (Put '1' in quotient)
    - Copy down next bit
  - If first bit is '0'
    - subtract '000'
    - (Put '0' in quotient)
    - Copy down next bit
- Append remainder to data as checksum

```
         10110
    101 |1001100
         101
          011
          000
           111
           101
            100
            101
             010
             000
              10
```

$n = 2$

$P = 101$

$M = 10011$

$M' = 1001110$



## CRC Control at Receiver

- Divide received data with *P*
- If remainder is '00', data is OK
  - Strip off CRC bits
- Otherwise discard data

```
         10110
    101 |1001110
         101
          011
          000
           111
           101
            101
            101
             000
             000
              00
```

$n = 2$

$P = 101$

$M = 10011$

$M' = 1001110$



## Generator Polynomials

- Binary numbers can be represented as polynomials
  - (CRC is a called polynomial code checksum)
  - Bit value is coefficient of a term
  - Exponent indicates the bit position, starting at 0
  - Example: 100111 $\Rightarrow$

$P(X) = 1 \times X^5 + 0 \times X^4 + 0 \times X^3 + 1 \times X^2 + 1 \times X + 1 \times X^0$

$P(X) = X^5 + X^2 + X + 1$

- Standard polynomials

ITU-16: $X^{16} + X^{12} + X^5 + 1$

ITU-32: $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$



## CRC

- Effective error detection
  - All burst errors that affect an odd number of bits
  - All burst errors of length less than or equal to degree of polynomial
  - With high probability longer errors
- Simple implementation in hardware
  - Shift register circuit
  - CRC often appended to the data (trailer)

## Checksum

- Treat the data as a sequence of integer numbers in binary format
  - Divide data into *k* units, with *n* bits in each
  - Compute the sum of all *k* units using ones complement arithmetic
  - Complement the sum and append the result to the data
- Receiver
  - Compute the sum over the data
  - Complement the sum
  - If the result equals zero, the data is accepted (otherwise rejected)



## Checksum

- Less effective than CRC
  - Easier to implement in software
- Detects
  - all errors involving an odd number of bits
  - Most errors involving an even number of bits
    - Two opposite bit inversions may balance out each other



## Correction of Errors

- Forward Error Correction (FEC)
  - Error-correcting codes
  - Replace CRC, checksum etc with a code that can automatically correct the error
  - Needs more redundancy bits
- Retransmission (ARQ)
  - Can be used both for bit errors and frame loss
  - A frame with bit errors is dropped (lost)

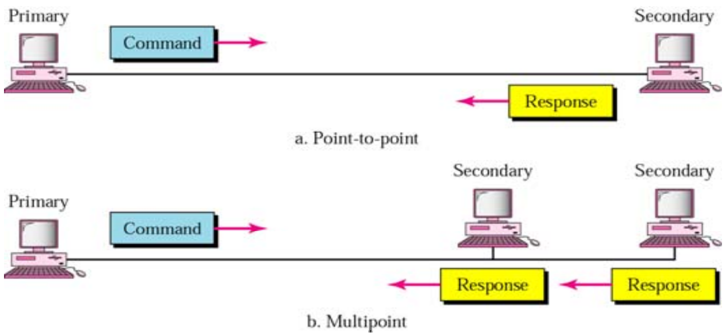


## Data Link Example: HDLC

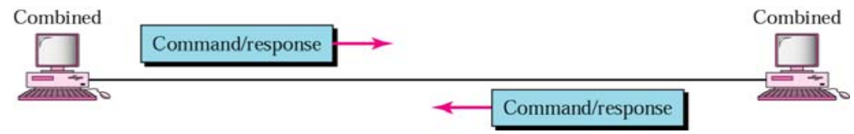
- High-level Data Link Control
- Half-duplex and full-duplex
- Point-to-point and multipoint links
- Normal response mode (NRM) and asynchronous balance mode (ABM)

# HDLC Normal Response Mode

- Unbalanced
- Point-to-point and multipoint links




# HDLC Asynchronous Balanced Mode



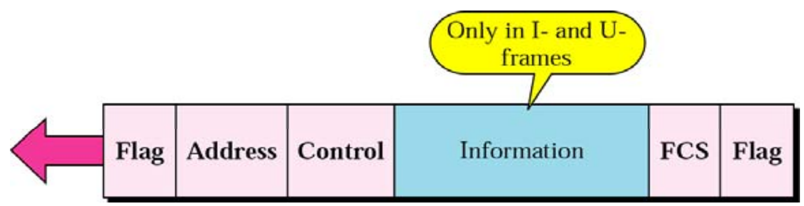



# Three HDLC Frame Types

- Information frames (I-frames)
  - User data
  - Acknowledgements
    - Piggybacking
- Supervisory frames (S-frames)
  - Control information related to user data
    - RR—Receive Ready (ACK)
    - RNR—Receive not Ready (ACK, receiver busy)
    - REJ—Reject (REJ)—(NACK, Go-back-N)
    - SREJ—Selective Reject (NACK, Selective-repear ARQ)
- Unnumber frames (U-frames)
  - System management
    - Link setup and tear-down
      - Setting transmission mode, etc



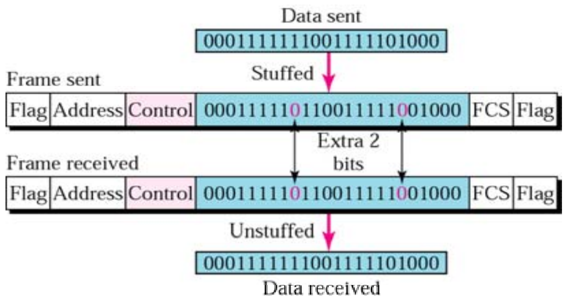
# HDLC Frame Format




- Flag
  - Start and end
  - Binary 01111110
    - $7E_{16}$
- Address
  - Of secondary
- Control
- Information
  - User data
  - Management information
- FCS field
  - 2- or 4-byte CRC

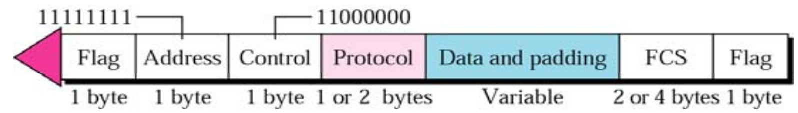
## HDLC Bit Stuffing

- Data may contain flag pattern 01111110
- Sender: insert ("stuff") an extra 0 after five 1s
- Receiver: remove 0 after five 1s




## Data Link Example: (PPP)

- Point-to-point Protocol
- Control and management of data transfer over physical (point-to-point) links
  - Dedicated link with two stations
  - Traditional modem, DSL, etc
- Based on HDLC frame format

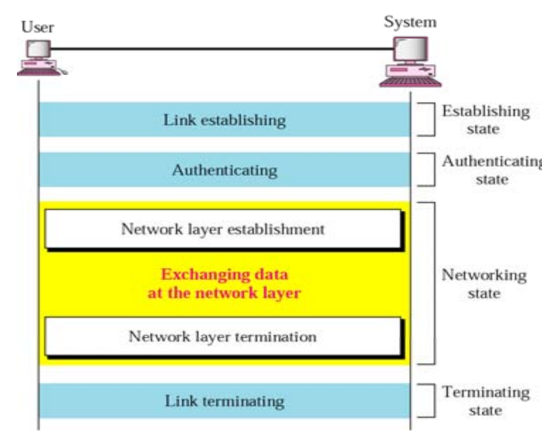



## PPP Protocol Family

- Link Control Protocol (LCP)
  - Establish, disconnect link
  - Negotiate options—maximum receive unit, authentication, compression
- Authentication
  - Password Authentication Protocol (PAP)
  - Challenge Handshake Authentication Protocol (CHAP)
- Network Control Protocol (NCP)
  - Internetwork Protocol Control Protocol (IPCP)



## PPP Example



## Summary

- Flow control
  - Stop and wait
  - Sliding window
- Bit error detection
  - Parity control
  - Checksum
  - Cyklic redundancy check (CRC)
- Detecting frame loss: sequence numbers
- Error control: retransmision (ARQ)
  - Stop and wait ARQ
  - Go-back-N ARQ
  - Selective reject ARQ
- Two examples:
  - HDLC
  - PPP



## Reading Instructions

- Behrouz A. Forouzan, "Data Communications and Networking,"
  - 10 Error Detection and Correction
    - 10.1 Types of errors
    - 10.2 Detection
    - 10.3 Error correction
  - 11 Data Link Control and Protocols
    - 11.1 Flow and error control
    - 11.2 Stop-and-wait ARQ
    - 11.3 Go-back-*N* ARQ
    - 11.4 Selective Repeat ARQ
    - 11.5 HDLC
  - 12 Point-to-point access: PPP