UNIT – 2

The **Data Link Layer** handles reliable communication between directly connected devices. Here are its key design points:

Functions:

1. **Service Interface to Network Layer**: Ensures smooth communication with the network layer.
2. **Error Handling**: Detects and corrects transmission errors.
3. **Flow Control**: Manages data flow so fast senders don't overwhelm slow receivers.

Frames:

- Data from the network layer is wrapped into **frames**, which include a header, payload (data), and trailer for transmission.

Services Offered:

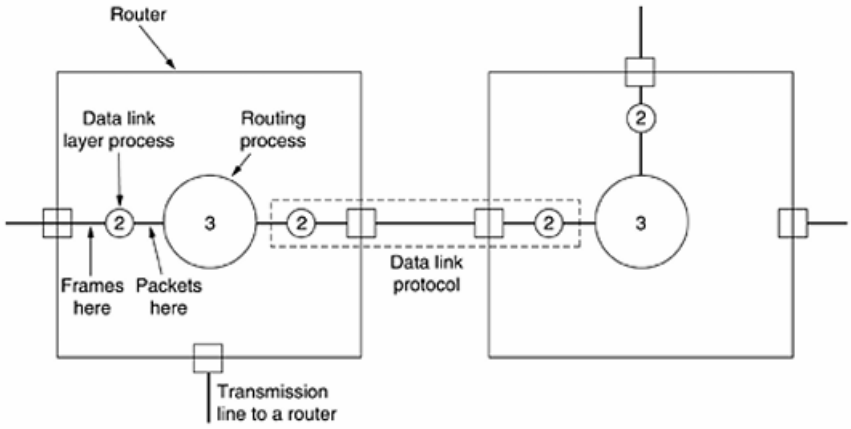
1. **Unacknowledged Connectionless**: Simple sending of frames without acknowledgments; used in error-free or real-time systems like voice communication.
2. **Acknowledged Connectionless**: Each frame is acknowledged for reliability; suitable for unreliable channels like wireless.
3. **Connection-Oriented**: Ensures frames are delivered once, in the correct order, and with error checks.

Process:

- Frames are sent and received with mechanisms to manage errors and flow.
- On unreliable channels (e.g., wireless), acknowledgment and retransmission improve reliability.
- On reliable channels (e.g., fiber), simpler protocols are sufficient.

In a network, the data link layer helps routers or devices manage errors and ensures packets are passed accurately between devices, making communication efficient and robust.

Figure 3-3. Placement of the data link protocol.

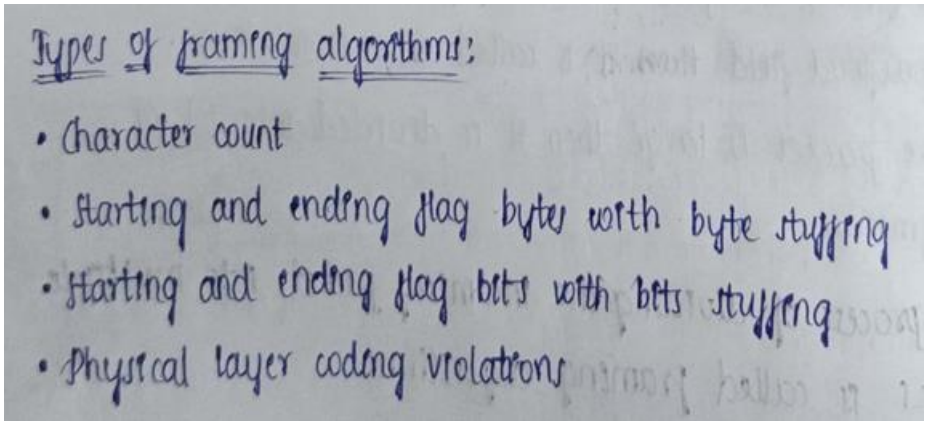


Framing in the Data Link Layer

The **Data Link Layer** splits a continuous bit stream from the Physical Layer into smaller, manageable **frames** for transmission. It ensures data integrity by detecting and, if needed, correcting errors during transmission.

Why Framing?

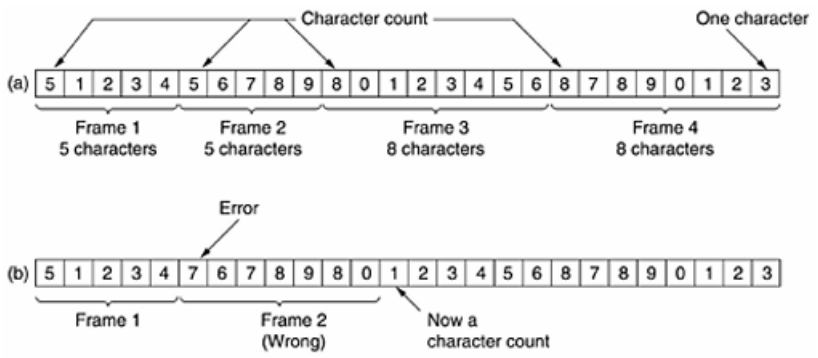
- The raw bit stream may contain errors or lose synchronization.
- Frames help organize the data, making it easier to detect and handle errors.
- Each frame contains a header, payload (data), and a trailer for error checking.

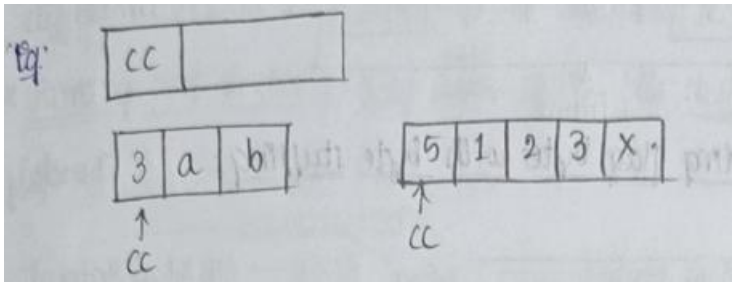


Methods of Framing:

1. **Character Count**:
   - The header specifies the number of characters in the frame.

Figure 3-4. A character stream. (a) Without errors. (b) With one error.





   - Due to error in one frame, the next frames are also damaged and corrupted.
   - Thus, interframe dependency is there in the character count, thus it is not a good solution.
   - **Problem**: If the count is corrupted, synchronization is lost. Rarely used.
2. **Flag Bytes with Byte Stuffing**:

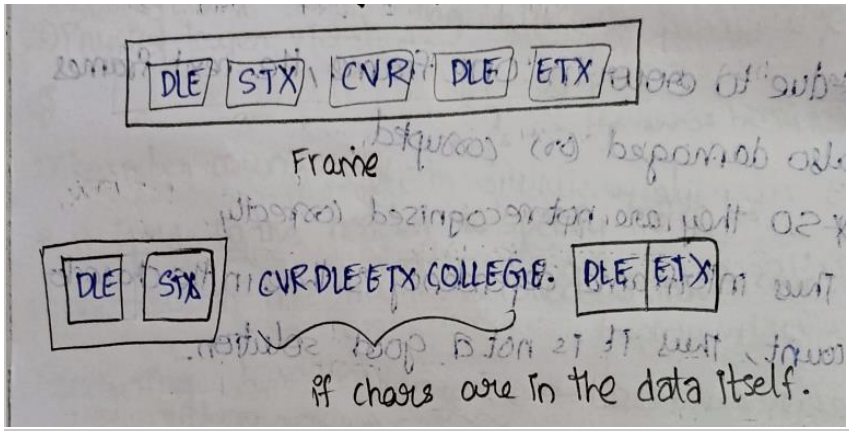
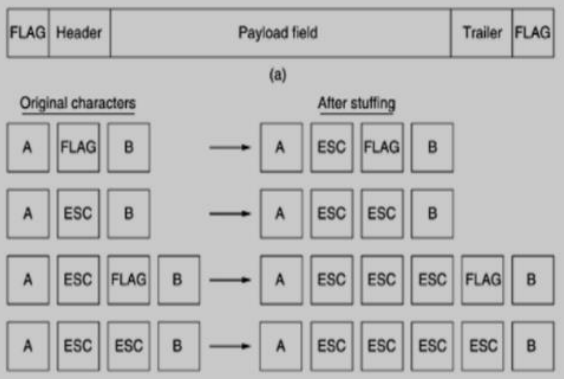


Figure 3-5. (a) A frame delimited by flag bytes. (b) Four examples of byte sequences before and after byte stuffing.



- Frames start and end with a special **flag byte (DLE STX , DLE ETX)**
- If the flag byte(DLE ETX) appears in the data, an **escape byte(DLE)** is added before it.
- Issue: Tied to 8-bit characters; not suitable for all systems.

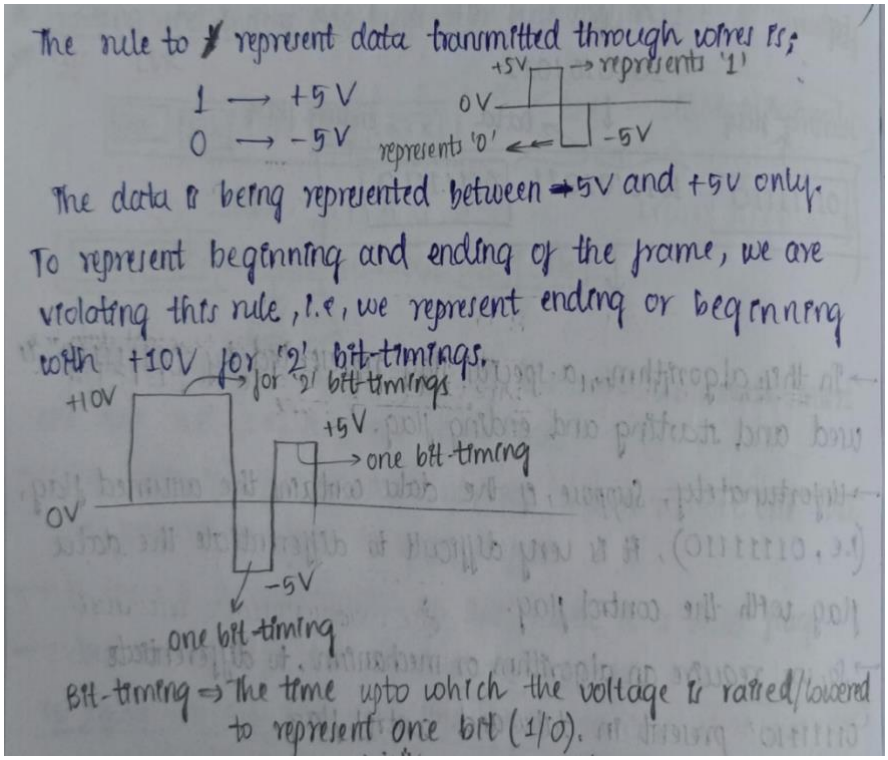
3. **Bit Stuffing**:

- Frames are marked by a special bit pattern (01111110).
- If the data contains the assumed flag(01111110), then it is very difficult to differentiate the data flag with the control flag.
- **Sender logic** : If data contains five consecutive 1s, a 0 is added to prevent confusion.
- **Receiver logic:** if there are five ones followed by a zero then removes it. Works for arbitrary-sized frames.

Figure 3-6. Bit stuffing. (a) The original data. (b) The data as they appear on the line. (c) The data as they are stored in the receiver's memory after destuffing.

4. **Physical Layer Coding Violations**:

- Some networks use special signal patterns to mark frame boundaries. – like if +10v and 2 bit time, it is starting bit and +5v is 1 and -5v is 0 as always.
- Example: High-high or low-low signals.



Error Control:

Error control refers to the process of handling errors that occur while transmitting data from the source to the destination. It is crucial in ensuring the reliability of communication systems.

Components of Error Control

1. **Error Detection:** Identifies the presence of errors in the transmitted data.
2. **Error Correction:** Deals with fixing the detected errors to ensure accurate communication.

Error Correction Mechanisms

Error correction is divided into two approaches:

1. **Correction by the Destination (Self-Healing Solution):** The destination resolves errors without requiring feedback from the source.
2. **Correction by Feedback from the Destination to the Source:** The feedback can take the form of acknowledgments. These acknowledgments are classified as:
   - **Positive Acknowledgment (ACK):** Indicates error-free data reception.
   - **Negative Acknowledgment (NACK):** Indicates errors in the received data.

Steps in Error Correction with Feedback

- **If ACK is positive:** The data is considered error-free.
- **If NACK is received:** Retransmission policy is applied, and the data is sent again.

Challenges in Retransmission

1. If retransmitted data arrives at the destination multiple times, duplicate frames/packets can occur.

2. To address this issue, sequence numbers are used to uniquely identify frames and avoid duplication.

RTT (Round Trip Time)

- RTT is the time taken to transmit data from the source to the destination and receive feedback or acknowledgment.
- Problem:
  If an acknowledgment itself is damaged or lost, the source may enter an infinite waiting state. To avoid this, a timer is introduced.

Timers in Error Control

- A timer starts for every specific frame with the RTT value.
- If the timer expires and no acknowledgment is received, retransmission is triggered automatically.

ARQ (Automatic Repeat Request)

At the data link layer, protocols that involve:

- Acknowledgments (ACK/NACK),
- Retransmission,
- Sequence numbers, and
- Timers
  are collectively referred to as ARQ protocols.

Flow Control

Flow control regulates the flow of data between two devices with differing transmission speeds to ensure smooth communication.

Types of Flow Control Approaches

1. **Feedback-Based Flow Control:**
   - Involves feedback from the receiver to the sender to adjust the data flow.
   - Example: *Stop-and-Wait Protocol.*
2. **Rate-Based Flow Control:**
   - Relies on pre-determined flow rates to control data transmission.
   - Example: *Flow Negotiation Protocol. - SLA*

Error Detection and Error Correction

Error detection identifies errors during the transmission of data.

Types of Transmission Errors

1. **Single-Bit Errors:** Occur in serial communication.

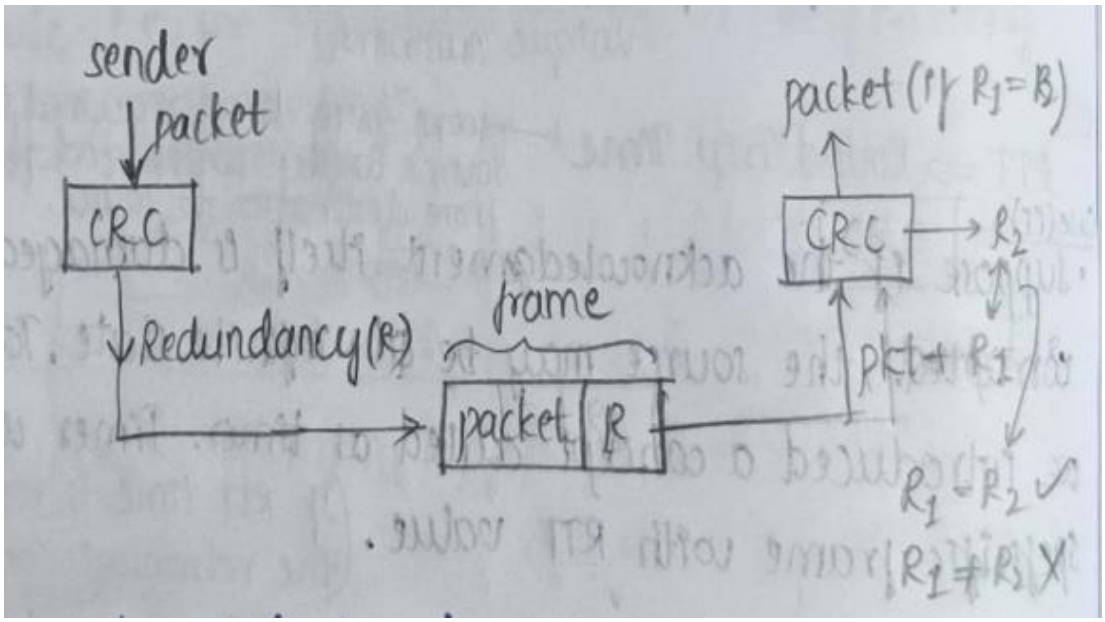
2. **Multi-Bit Errors:** Occur in parallel communication.

Mechanism of Error Detection

- Errors are detected using the Exclusive OR (XOR) operation.
- The original string and the received string are compared bit by bit:
  - If the result is 0 for all bits, it means no errors.
  - If the result is non-zero, it indicates errors are present.

Modules of Error Detection

1. **Sender:** Performs initial encoding and transmission.
2. **Receiver (e.g., CRC):** Detects errors using techniques like Cyclic Redundancy Check (CRC).



Cyclic Redundancy Check (CRC) is an error-detecting code used to check the integrity of data during transmission or storage. It involves dividing the data by a fixed divisor (generator polynomial) and appending the remainder (checksum) to the data. The receiver performs the same check, and if the remainder matches, the data is considered error-free.

CRC Algorithm: -sender:-

S1:- let d(x) be a data word from the network layer

S2:- The given generator function (agreed by source and destination), so g(x)

S3:- Find the length of the generator function (the binary string length say 'L').

S4:- Append 'L-1' binary zeroes to the data word d(x)

S5:- Implement or start arithmetic binary division (CRC Algorithm)

S6:- By the end of the division we will get a remainder i.e "Redundancy" - R(x)

S7:- The final transmitted PDU (Protocol Data Unit) is code word - C(x) = d(x) + R(x).

-Receiver:

S1:- The received code word is C(x)

S2:- Already the agreed/committed the generator function g(x).

S3:- Apply a arithmetic binary division g(x) as a divisor and C(x) as a dividend.

S4:- By the end of the division operation you will get a remainder say (S(x)); Syndrome.

S5:- If the syndrome S(x) ≠ 0, then there is no error in the received data C(x).

If S(x) is non-zero there is a error in the codeword C(x).

NOTE:- The arithmetic binary division here is

Modulo-2 division

↓

XOR operation.

Eg:- d(x) = 10101111 G(x) = 1011

Append (4-1) = '3' zeroes to D(x)

10101111000 ⇒ auxiliary codeword

1011) 10101111000 (

1011

0001111

1011

01000

1011

00111

1011

0111

R(x)

codeword is C(x) = 10101111111

d(x) R(x)

Receiver routine:-

1011) 10101111111 (

1011

0001111

1011

01001

1011

001011

1011

0000

case(i):- Suppose, one of the bit is inverted (changed from '0' to '1' (or) '1' to '0') in the codeword,

prove that there is an error in the received data.

The actual codeword is 1010111111

According to the question, let us assume that the received codeword is 1000111111 (higher order 3rd bit is inverted).

1011) 1000111111 (

1011

001111

1011

01001

1011

001011

1011

00000 ← S(x)

∴ the codeword received has errors.

12/8/24

#Polynomial CRC

- The conversion from polynomial to the binary:-

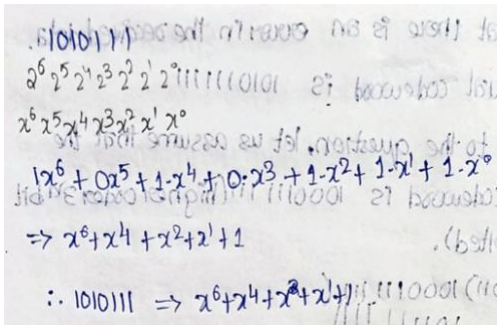
Eg:- $x^6+x^5+x^3+x^2+1$

$\Rightarrow 1\cdot x^6+1\cdot x^5+0\cdot x^4+1\cdot x^3+1\cdot x^2+0\cdot x^1+1\cdot x^0$

$\therefore x^6+x^5+x^3+x^2+1 \Rightarrow 1101101$

- Conversion from binary to polynomial:-

Eg:- 1010111



Elementary Data Link Layer Protocols

Data link layer protocols manage the data transfer between network devices. They can be classified into two main types:

1. **Data link layer Protocols for Noiseless Channels:** These protocols assume the communication medium is error-free.
2. **Data link layer Protocols for Noisy Channels:** These are designed to handle errors in data transmission.

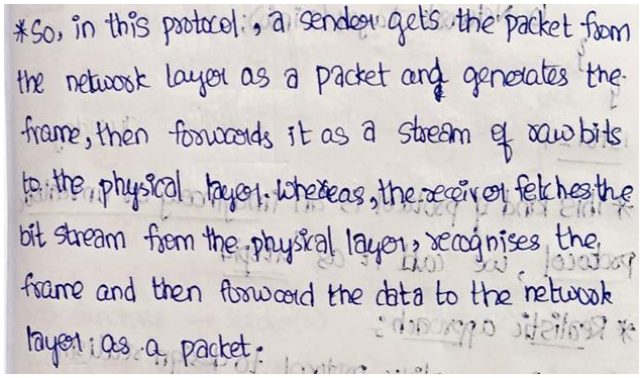
<u>Data Link Layer Protocols for Noiseless Channels:</u>

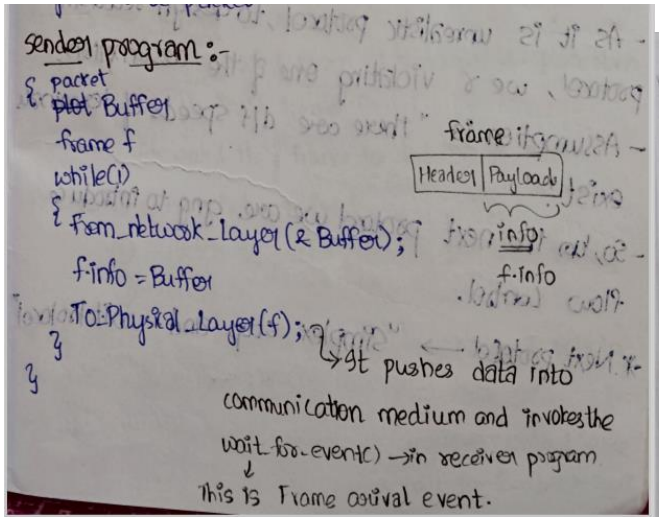
Network model assumptions

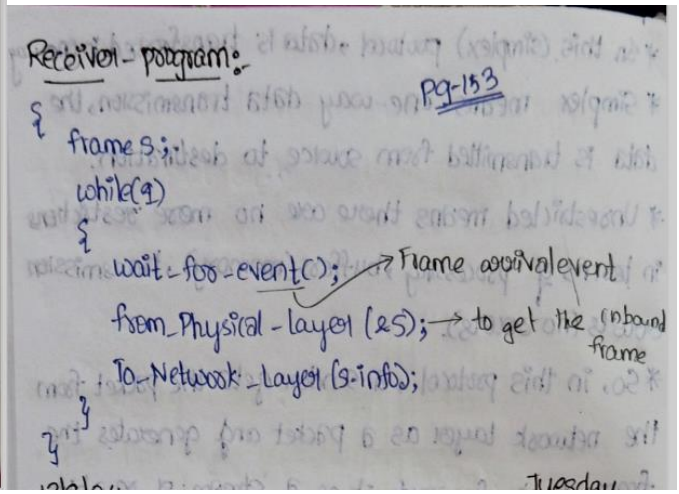
- There are 'n' number of stations/terminals in the network.
- Each station:
  - Has data ready to transmit.
  - Is ready to receive data from others.
- Error-free communication medium: No errors occur during data transfer.
- Each machine has an infinite buffer for storing data temporarily.
- All systems are high-end terminals and operate at equal speed.
  - Example: FM Radio.

1. **<u>Simplex Unrestricted Protocol:</u>**

- This is the first protocol designed for noiseless channels.
- Simplex: One-way data transmission (e.g., sending only from source to destination).
- Unrestricted: No limitations regarding: Processing speed, Memory (buffer size), Transmission errors (assumed to be zero).







- This protocol is an **imaginary or unrealistic protocol**, referred to as "Utopia."
- Since "Utopia" is unrealistic, designing a realistic protocol involves violating one of its assumptions.
- **Assumption:** "There are different speeds of terminals."
- Thus, to address this issue, **Flow Control** is introduced.
- The next protocol discussed is the **Simplex Stop-and-Wait Protocol.**

2. **<u>Simplex Stop-and-Wait Protocol</u>**

- A simpler protocol where the sender transmits data, and the receiver acknowledges receipt.
- Assumes **error-free communication media.**
- The receiver has limited **buffer capacity** and **processing speed.**
- The protocol prevents the sender from overwhelming the receiver with data faster than it can process.

Steps in the Protocol:

1. The sender (S) sends a frame (f1) to the receiver (R).
2. The receiver receives the frame and processes it.
3. The sender waits for an acknowledgment (ACK) from the receiver before sending the next frame.

Functions of Components:

1. **Sender (S):**
   - Fetches packets from the network layer and places them into frames.
   - Sends the frames to the receiver one at a time.
   - After sending, the sender waits for an acknowledgment from the receiver.
2. **Receiver (R):**
   - Receives packets, identifies the boundaries of the data, and forwards them to the upper layer.

- Since the receiver processes data slower than the sender transmits, sending too many packets can lead to **overflow errors.**
- The **stop-and-wait mechanism** ensures that after sending a frame, the sender waits for feedback (ACK) from the receiver before sending the next frame.

```
typedef enum {frame_arrival} event_type;
#include "protocol.h"

void sender2(void)
{
 frame s;
 packet buffer;
 event_type event;

 while (true) {
      from_network_layer(&buffer);
      s.info = buffer;
      to_physical_layer(&s);
      wait_for_event(&event);
 }
}

void receiver2(void)
{
 frame r, s;
 event_type event;
 while (true) {
      wait_for_event(&event);
      from_physical_layer(&r);
      to_network_layer(&r.info);
      to_physical_layer(&s);
 }
}
```

Data Link Layer Protocols for Noisy Channels:

1. A Simplex Protocol for a Noisy Channel (Stop and wait):

This protocol ensures that data is sent reliably over a channel that may lose or damage frames. It uses **sequence numbers** to avoid duplicate frames and make communication error-free.

1. **Problem**: If a frame is lost or damaged, the sender might resend it. Without a way to identify duplicates, the receiver might deliver the same data twice.
2. **Solution**:
   - Each frame gets a **sequence number** (0 or 1).
   - The receiver checks the sequence number to know if the frame is new or a duplicate.
   - After receiving the correct frame, the receiver updates its sequence number.
3. **How It Works**:
   - The sender sends a frame and waits for an acknowledgment (ACK) from the receiver.
   - If no ACK is received within a set time (because of loss or error), the sender resends the frame.
   - The process continues until the receiver gets the correct frame.
4. **Key Idea**:
   - The receiver only accepts a frame if it matches the expected sequence number. Duplicates are rejected.

2. Sliding Window Protocols Simplified

What is a Sliding Window Protocol? Sliding Window Protocols allow data to flow in both directions on the same communication channel, avoiding the inefficiency of using separate channels for data and acknowledgments. This protocol uses a "window" to control how many frames (units of data) can be sent or received at a time.

How It Works

1. **Two-Way Communication:**
   - Instead of using two separate channels (one for sending data and another for receiving acknowledgments), the same channel is used for both.
   - Data frames and acknowledgment frames are interleaved, improving channel efficiency.
2. **Piggybacking:**
   - Acknowledgments are added to outgoing data frames instead of sending them separately.
   - This reduces overhead and makes better use of bandwidth.
3. **Sequence Numbers:**
   - Frames are numbered to keep track of which have been sent and acknowledged.
   - The numbers cycle within a range, typically determined by the size of the sequence number field.

- **Sender's Window:** Contains frames that have been sent but not yet acknowledged.
- **Receiver's Window:** Contains frames the receiver is willing to accept.

The size of these windows can vary:

- **Fixed size:** Easier to manage but less flexible.
- **Variable size:** Adapts to network conditions but adds complexity.

Types of Sliding Window Protocols

1. Stop-and-Wait Protocol (One-Bit Sliding Window):

- Only one frame is sent at a time. The sender waits for an acknowledgment before sending the next frame.
- Simple but inefficient for long delays (e.g., satellite communication).

Figure 3-13. 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.

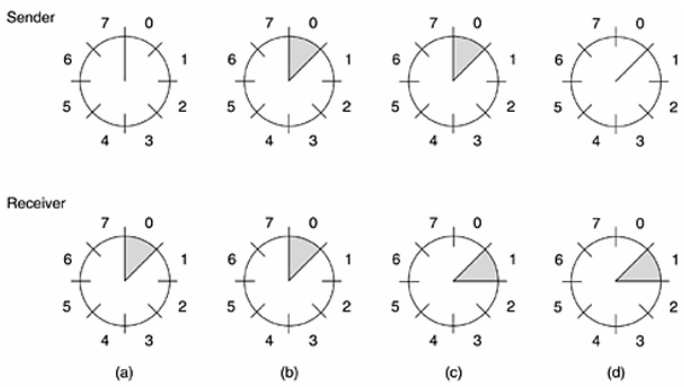
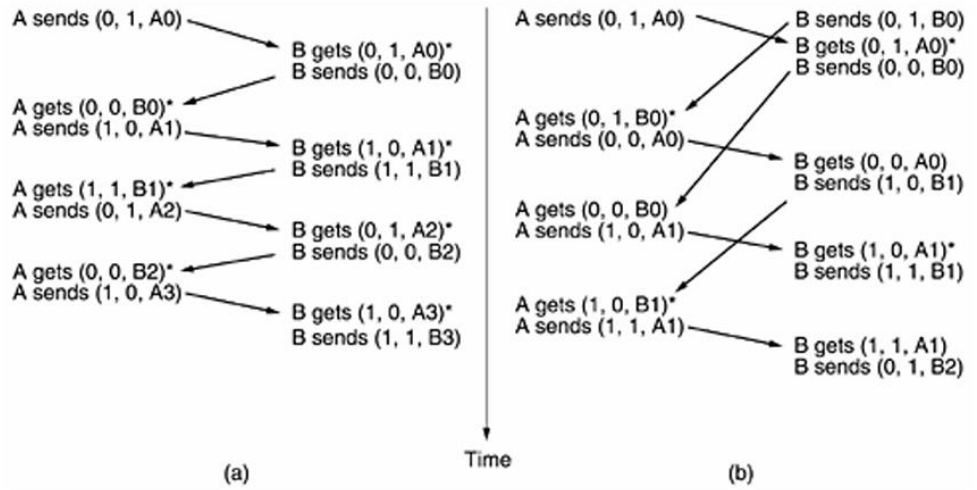


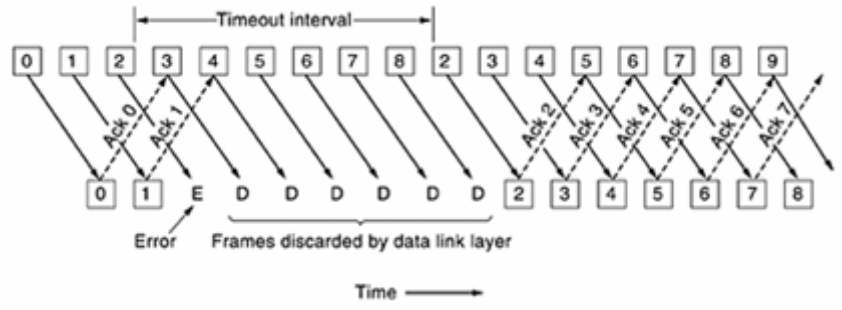
Figure 3-15. Two scenarios for protocol 4. (a) Normal case. (b) Abnormal case. The notation is (seq, ack, packet number). An asterisk indicates where a network layer accepts a packet.



2. Go-Back-N Protocol:

The Go-Back-N (GBN) protocol is a data link layer protocol used to ensure reliable data transfer over a noisy channel. It employs a sliding window mechanism to manage the flow of packets between the sender and receiver.

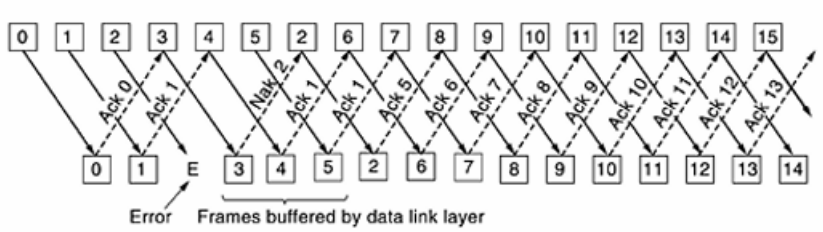
- **Initial Packet Transmission**:
  - The sender transmits the 0th packet first to the data link layer.
  - Following this, packets numbered **1, 2, 3, ...** are sent sequentially without waiting for individual acknowledgments.
- **Acknowledgment Mechanism**:
  - After some time, the receiver sends a **cumulative acknowledgment (ACK)**.
  - For example, if packets 0–2 are received successfully, an acknowledgment such as **"ACK 3"** is sent to indicate all packets up to number 2 have been received.
- **Handling Errors**:
  - Suppose an error occurs in packet **2**.
  - The data link layer does **not send any acknowledgment** for packets after this, as an error occurred (no selective acknowledgment is used in this protocol).
  - The network layer continues sending packets **3, 4, 5, 6, 7, 8**, but the data link layer discards them since packet **2** is missing.
- **Timeout and Retransmission**:
  - When a **timeout event** occurs for packet **2**, the sender **retransmits packet 2**.
  - Subsequently, the sender also retransmits all subsequent packets (**3, 4, 5, 6, 7, 8**) as these were also discarded by the receiver.



3. Selective Repeat Protocol:

The Selective Repeat (SR) protocol improves on the inefficiency of Go-Back-N by selectively retransmitting only the packets that are corrupted or lost.

- **Error Handling**:
  - Suppose an error occurs in packet **2**.
  - The receiver sends a **negative acknowledgment (NACK 2)** to indicate that packet 2 was corrupted.
- **Buffering Mechanism**:
  - All correctly received packets after the erroneous one (**packets 3, 4, 5**) are stored in a **buffer** at the receiver.
  - These packets are not discarded as in Go-Back-N.
- **Retransmission**:
  - The sender retransmits **packet 2** after receiving the NACK.
  - Once packet **2** is successfully received, a **cumulative acknowledgment (ACK)** is sent, indicating that all buffered packets (**2, 3, 4, 5**) are now processed correctly.
- **Acknowledgment Handling**:
  - For any packet received correctly, individual ACKs or cumulative ACKs are sent as per the acknowledgment logic.



Comparison of Go-Back-N and Selective Repeat

Feature	Go-Back-N	Selective Repeat
Retransmission	All packets after error	Only erroneous packets
Efficiency	Moderate	High
Buffer Requirement	Minimal	High
Acknowledgments	Cumulative ACK only	Both ACK and NACK
Use Case	Low error rate scenarios	High error rate scenarios

Example of Data link layer protocol:

HDLC (High-Level Data Link Control)

- A **bit-oriented protocol** that ensures reliable communication at the data link layer. It's used for framing, error control, and sequencing.
- **Frame Structure**:

1. **Flags**: Mark the start and end (01111110).
2. **Address**: Receiver identifier (useful in networks).
3. **Control**: Manages data flow and errors.
4. **Data**: Actual message.
5. **Checksum**: Detects errors.

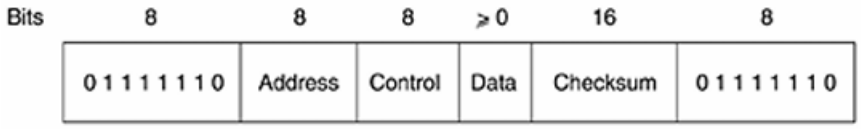
- **Types of Frames**:

1. **Information-frames**: Carry user data.
2. **Supervisory - frames**: Handle acknowledgments and flow control.
3. **Unnumbered - frames**: For setup/reset tasks.

- **Key Features**:

1. Uses **sliding window** for flow control (up to 7 frames).
1. Handles errors by retransmitting only corrupted frames.

Figure 3-24. Frame format for bit-oriented protocols.

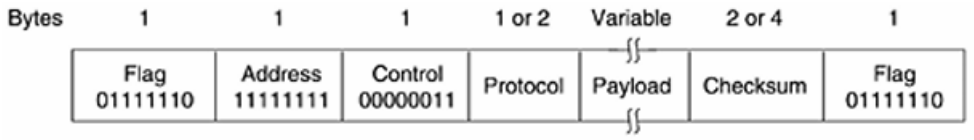


PPP (Point-to-Point Protocol)

- A protocol for **direct connections** (e.g., home to ISP). It's flexible and supports multiple network protocols like IP.
- **Frame Structure**:

1. **Flag**: Marks start/end of the frame.
2. **Address**: Broadcast address (11111111).
3. **Control**: Simplifies flow control (00000011).
4. **Protocol**: Identifies the type of data (e.g., IP).
5. **Data**: Message content.
6. **Checksum**: Error detection.

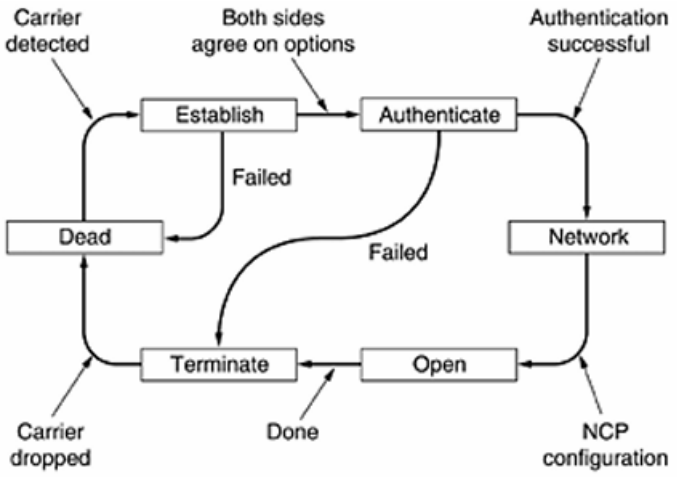
Figure 3-27. The PPP full frame format for unnumbered mode operation.



- **Phases**:

1. **Establish**: Sets up the connection.
2. **Authenticate**: Verifies identity (optional).
3. **Network**: Configures IP or other protocols.
4. **Data Transfer**: Sends/receives data.
5. **Terminate**: Ends the connection.

Figure 3-28. A simplified phase diagram for bringing a line up and down.



Part - 2

MAC - The Medium Access Control

- In broadcast networks, multiple users compete to use a single communication channel.
- Coordination is needed to avoid collisions (like multiple people talking at the same time on a conference call).
- This challenge is addressed by Medium Access Control (MAC) protocols, part of the data link layer.

- **Purpose**: Manages access to a shared communication channel in networks.
- **Example**: In Ethernet (wired LAN), the MAC sublayer determines how devices transmit data without interfering with each other.

<u>The Channel Allocation Problem</u>

Static vs. Dynamic Channel Allocation

1. Static Channel Allocation

- **Frequency Division Multiplexing (FDM)**.
  - FDM divides the channel into **frequency bands**, and each user gets a dedicated frequency to transmit data.
  - No interference between users as each has a separate frequency.
  - If a user isn't sending data, their frequency remains unused, wasting bandwidth.
  - **Real-life example**: Radio stations (FM 90.1, 94.5, etc.).
  - Think of a **radio station**: Each station broadcasts on a unique frequency, and you tune in to listen. If one station stops broadcasting, that frequency remains unused.
  - FDM works well with a fixed number of users but becomes inefficient with bursty or variable traffic.
- **Time Division Multiplexing (TDM)**
- TDM divides the channel into fixed **time slots**.
- Each user is assigned a specific time slot to send data, whether they need it or not.
- All users share the same channel but use it at different times.
- Unused time slots go idle, leading to inefficiency.
- **Example** : Imagine 5 people taking turns to speak for 1 minute each on a single phone line. If one person has nothing to say, their 1 minute is wasted instead of being used by others.
- TDM is simple but inefficient for bursty traffic because idle slots waste bandwidth.
- Problems:
  - **Wasted Bandwidth:** Unused portions cannot be reassigned when some users are idle.
  - **Inefficiency:** Not suitable for bursty traffic (common in computer networks).
  - **Increased Delays:** Dividing the channel into subchannels increases delays for data transmission.

2. Dynamic Channel Allocation

- Allocates bandwidth on demand, adapting to traffic.
- More efficient for handling variable and bursty traffic.
- Example: **Cellular Networks** dynamically assign frequencies or time slots to active users.

Assumptions of Dynamic Channel Allocation

1. **Station Model**:
   - There are **N independent stations** (like computers or phones).
   - Each station generates data (frames) at a constant rate (λ).
   - A station waits (is blocked) until its data is successfully sent.
2. **Single Channel**:
   - Only **one communication channel** is available.

- All stations share the same channel for sending and receiving.

3. **Collision Assumption**:

- If two stations send data at the same time, a **collision** occurs, making the data unreadable.
- Collisions are detected, and the data is retransmitted later.

4. **Time Models**:

- **Continuous Time**: Data transmission can start at any moment.
- **Slotted Time**: Time is divided into slots; transmissions only begin at the start of a slot.

5. **Carrier Sense**:

- **With Carrier Sense**: Stations check if the channel is free before sending data.
- **Without Carrier Sense**: Stations send data without checking, leading to more collisions.

Multiple Access Protocols (Aloha, CSMA, CSMA/CD)

ALOHA Protocol

- ALOHA is a method to allow multiple devices (computers, phones, etc.) to share a single communication channel.
- It solves the problem of managing who can send data and when.

Types of ALOHA:

1. **Pure ALOHA (1, 2, 3, 4(a), 5(b))**:

- Devices send data **whenever they have something to send**, without checking if the channel is free.
- **Collisions** happen when two devices send data at the same time, and both data transmissions are destroyed.
- Devices detect collisions and retransmit their data after waiting for a **random time**.
- **Efficiency**: At most **18%** of the channel is successfully used because collisions are common.

Example of Pure ALOHA:

1. A computer sends a message to the channel immediately.
2. If another device also sends at the same time, a **collision** happens.
3. Both devices detect the collision and wait a random amount of time before retrying.

2. **Slotted ALOHA(1, 2, 3, 4(b), 5(b))**:

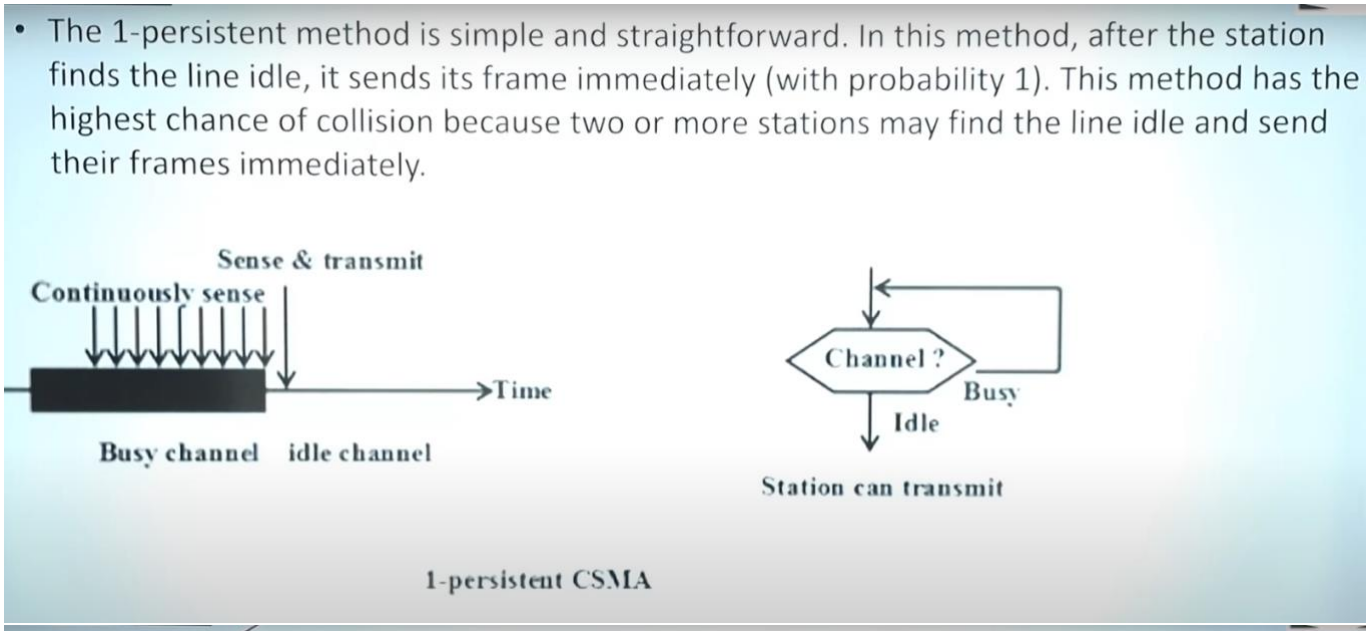
- Time is divided into **slots**. A device can only send data at the **start of a slot**.
- This reduces the chance of collisions because transmissions are better organized.
- **Efficiency**: At most **37%** of the channel is used successfully—better than pure ALOHA.

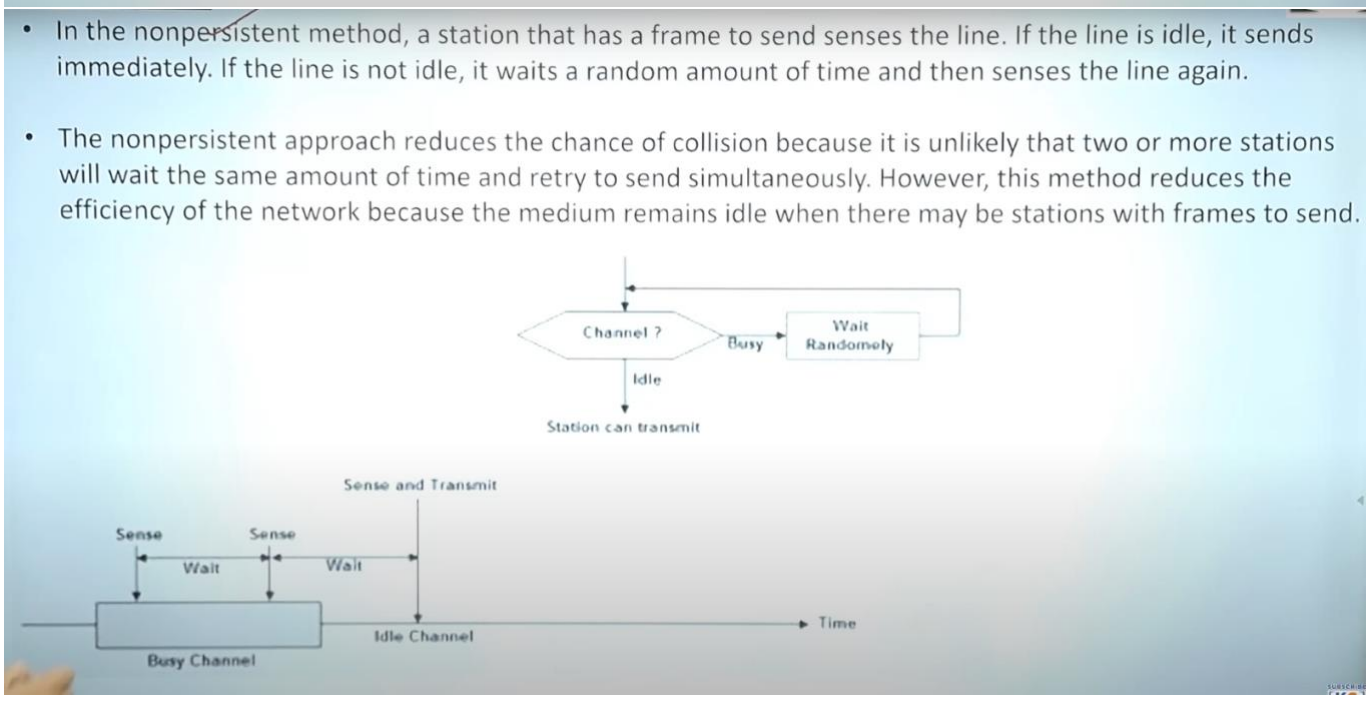
Example of Slotted ALOHA:

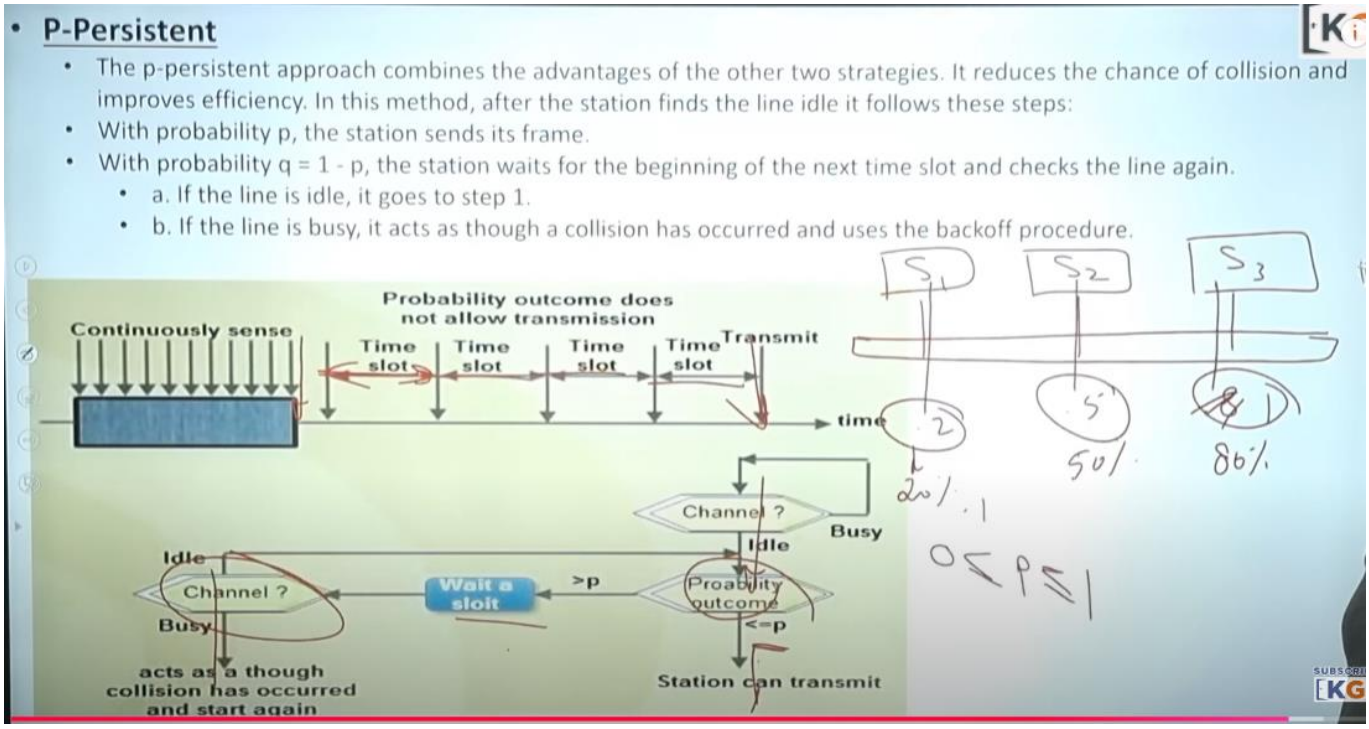
1. Devices are synchronized to send data at the **start of time slots**.
2. If two devices try to send in the same slot, a **collision** happens, and both wait randomly before retrying.

Carrier Sense Multiple Access (CSMA) Protocols

- CSMA protocols improve communication efficiency by letting devices "listen" to the channel before transmitting. – continuous waiting, context switching (notes)
- Unlike ALOHA, where devices send data anytime, CSMA reduces collisions by detecting if the channel is busy.







1. Persistent CSMA (1-Persistent):

- A device checks if the channel is idle.
- If idle: it immediately sends data (100% probability).
- If busy: it waits until the channel becomes idle and then transmits.
- Problem: If multiple devices try to send as soon as the channel is free, collisions can still occur.

Effect: Collisions are fewer than ALOHA but still possible, especially if propagation delays are significant (time taken for signals to travel across the network).

2. Nonpersistent CSMA:

- A device checks if the channel is idle.
- If idle: it sends data.
- If busy: it waits for a random time and checks again.

Effect:

- Reduces the chance of collisions since devices don't all try sending at the same time.
- Results in better channel utilization but increases the waiting time (delay).

3. P-Persistent CSMA (for Slotted Channels):

- The device only works in time slots.
- If the channel is idle:
  - Sends data with probability **p**.
  - Waits for the next time slot with probability **1 - p**.
- If the channel is busy: it waits for the next slot and repeats the process.

Effect: Balances delay and collisions, improving performance for slotted channels.

CSMA with Collision Detection (CSMA/CD)

- Improves over basic CSMA by detecting collisions during transmission.
- If two devices transmit at the same time, they both detect the collision quickly and stop transmitting.
- They wait for a random time before trying again.

Key Points:

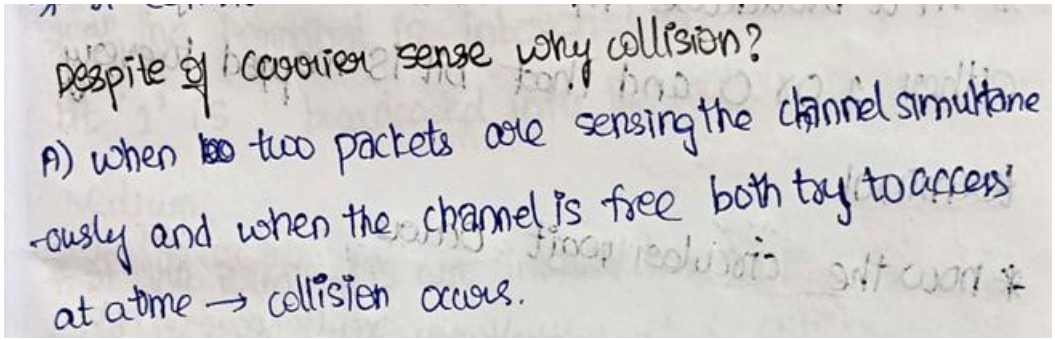
1. Devices listen to the channel while transmitting. If what they send doesn't match what they "hear," a collision has occurred.
2. Once a collision is detected, the transmission stops to save bandwidth.
3. Devices wait for **2τ** (twice the propagation time) to confirm the channel is clear before transmitting.

Real-Life Use:

- **CSMA/CD** is used in Ethernet networks, where devices share a cable to transmit data.

Limitation:

- CSMA/CD works only in **half-duplex mode** (cannot send and receive simultaneously).



Collision-Free Protocols – (bitmap, binary countdown)

Unlike CSMA/CD (Carrier Sense Multiple Access with Collision Detection), where collisions may occur during the contention period (when stations are trying to take control of the channel), collision-free protocols ensure no collisions occur at all, even during contention.

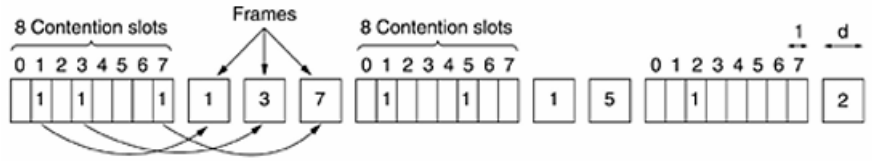
These protocols are designed for better efficiency, especially in systems with long propagation delays or short data frames.

1. Bit-Map Protocol

- Each station in the network is assigned a unique number (address) from 0 to N−1N-1N−1.
- The protocol divides the contention period into NNN slots, one for each station.
- Each station indicates its desire to send by transmitting a "1" in its slot. If it doesn't have anything to send, it remains silent (sends "0").
- After all slots are completed, every station knows which ones want to transmit.

- Stations then transmit in numerical order (based on their address) without collisions.
- (SAQ) Reservation protocols allow stations to reserve the channel before transmitting data, avoiding collisions during data transmission.

Figure 4-6. The basic bit-map protocol.



Scenario:

- There are **5 stations** (N=5): Station 0, Station 1, Station 2, Station 3, and Station 4.
- Each station is assigned a unique slot during the contention period.
- Stations **0, 2, and 4** have data to send.

Step-by-Step Process:

1. The contention period begins, and each station transmits "1" in its designated slot if it has data to send:
   - **Slot 0**: Station 0 sends "1" (has data).
   - **Slot 1**: Station 1 sends "0" (no data).
   - **Slot 2**: Station 2 sends "1" (has data).
   - **Slot 3**: Station 3 sends "0" (no data).
   - **Slot 4**: Station 4 sends "1" (has data).

The result is a **bit map**: **1 0 1 0 1**.

2. After the contention period, all stations know which ones want to send (Stations 0, 2, and 4).
3. Stations transmit their data **in numerical order**:
   - Station 0 transmits first.
   - Station 2 transmits second.
   - Station 4 transmits last.

Efficiency:

- If each frame has d bits, the overhead for the contention period is N bits.
- Efficiency = d/(d+N)

Advantages:

- No collisions.
- Clear, orderly transmission schedule.

Drawbacks:

- Overhead increases with the number of stations (N), as each contention period requires N bits, making it inefficient for large networks.

2. Binary Countdown Protocol

- Instead of reserving a bit slot for every station, stations use their **binary addresses** (e.g., 1010 for station 10) to compete for the channel.
- When a station wants to send data, it broadcasts its binary address bit by bit.
- All stations OR their bits together at each step (e.g., 0 + 1 = 1).
- If a station notices its bit is "overwritten" by a higher-priority bit (a 1), it gives up for that round. This means stations with higher binary addresses have priority.

Example: If four stations with addresses **0010**, **0100**, **1001**, and **1010** are trying to send:

1. In the first round, all stations send their first (most significant) bit.
   - 0,0,1,10, 0, 1, 10,0,1,1 → Results in a 1 (OR operation).
   - Stations **0010** and **0100** give up because they see higher-priority stations are competing.
2. In the next round, stations **1001** and **1010** send their next bit.
   - 0,00, 00,0 → Both continue since no conflict occurs.
3. In the third round, they send their next bit.
   - 1,01, 01,0 → Station **1001** gives up.
4. Station **1010** wins and transmits its data.

Advantages:

- Much more efficient than the Bit-Map Protocol for large networks, as only log⁡2N\log_2 Nlog2N bits are required per contention period.
- Can prioritize stations based on their addresses.

Drawbacks:

- Higher-priority stations (those with larger addresses) may dominate the channel, potentially leading to unfairness.

Limited-Contention Protocols

Limited-contention protocols combine the low delay of contention methods (like CSMA) at low load and the high efficiency of collision-free methods at high load. They dynamically group stations to control competition:

- **Low load**: More stations compete, minimizing delay.
- **High load**: Fewer stations compete, reducing collisions.

Definitions

- **Contention Protocols**: Stations compete for access (e.g., CSMA).
- **Collision-Free Protocols**: Avoid collisions using pre-assigned slots.
- **Limited-Contention Protocols**: Adjust station grouping and slot use based on load.

Ethernet Cabling

Ethernet cabling connects devices in a network and uses various types of cables based on speed, distance, and requirements. Here are the main types:

Figure 4-13. The most common kinds of Ethernet cabling.

Name	Cable	Max. seg.	Nodes/seg.	Advantages
10Base5	Thick coax	500 m	100	Original cable; now obsolete
10Base2	Thin coax	185 m	30	No hub needed
10Base-T	Twisted pair	100 m	1024	Cheapest system
10Base-F	Fiber optics	2000 m	1024	Best between buildings

1. 10Base5 (Thick Ethernet)

- **Description**: Thick, yellow cable (like a garden hose).
- **Speed**: 10 Mbps.
- **Range**: Up to 500 meters.
- **Connections**: Uses "vampire taps" to attach devices.
- **Limitations**: Hard to install and maintain.

2. 10Base2 (Thin Ethernet)

- **Description**: Thin, flexible coaxial cable.
- **Speed**: 10 Mbps.
- **Range**: Up to 185 meters per segment.
- **Connections**: Uses BNC connectors for easier installation.
- **Limitations**: Can only handle up to 30 devices per segment and is prone to issues like loose connectors.

3. 10Base-T (Twisted Pair)

- **Description**: Uses twisted-pair cables, with each device connected to a central hub.
- **Speed**: 10 Mbps.
- **Range**: Up to 100 meters (or 200 meters with high-quality cables).
- **Advantages**: Easy to add/remove devices and troubleshoot.
- **Limitations**: Cable length is limited.

4. 10Base-F (Fiber Optic)

- **Description**: Uses fiber optic cables.
- **Speed**: 10 Mbps.
- **Range**: Up to 2 km.
- **Advantages**: High noise immunity, better security, ideal for long distances.
- **Limitations**: Expensive.

Network Topologies

1. **Linear**: One long cable connecting all devices.
2. **Spine**: Vertical backbone connects horizontal cables on each floor.
3. **Tree**: Branches connect multiple devices; prevents interference.
4. **Segmented**: Multiple cable segments linked by repeaters.

Repeaters

- **Function**: Boost signal strength over long distances.
- **Limitation**: No two devices should be more than 2.5 km apart or pass through more than 4 repeaters.

This setup balances speed, distance, and ease of maintenance for different scenarios.

Wireless LAN Protocols

Wireless LANs (WLANs) connect devices without cables, using radio signals. They allow devices to communicate on the move, like in offices or homes.

Figure 4-11. A wireless LAN. (a) A transmitting. (b) B transmitting.

Challenges in Wireless LANs

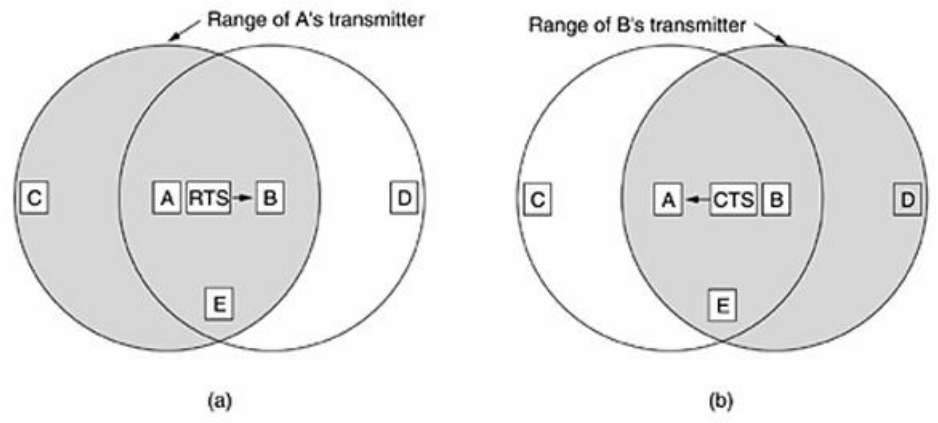
1. **Hidden Station Problem**: A device doesn't detect another device's signal and causes interference at the receiver.
2. **Exposed Station Problem**: A device unnecessarily avoids sending data due to nearby signals that won't interfere.

Key Protocols

1. CSMA (Carrier Sense Multiple Access)

- Devices listen for activity before sending data.
- **Limitation**: It checks for signals near the sender, not the receiver, which can still cause collisions.

Figure 4-12. The MACA protocol. (a) A sending an RTS to B. (b) B responding with a CTS to A.



2. MACA (Multiple Access with Collision Avoidance)

- Coordinates transmissions to avoid collisions:

1. **RTS (Request to Send)**: Sender asks the receiver for permission.
2. **CTS (Clear to Send)**: Receiver allows it and tells nearby devices to wait.
3. Sender transmits the data.

3. MACAW (MACA for Wireless)

- Improved version of MACA with:
  - Acknowledgment (ACK) after data is received.
  - Devices listen before sending an RTS.
  - Better fairness by managing backoff timers for each data stream.
  - Adjustments for congestion.

Benefits

- Prevents interference and collisions.
- Improves fairness and efficiency in wireless communication.