COMP 3721
Introduction to Data Communications

07b - Week 7 - Part 2

COMP 3721 数据通信导论

07b - 第7周 - 第2部分

Learning Outcomes

- By the end of this lecture, you will be able to
  - Explain the DLC (Data-Link Control) services.
  - Explain DLC protocols and how they function.

学习成果

- 在本讲座结束时，您将能够
  - 解释数据链路控制（DLC）服务。
  - 解释DLC协议及其工作原理。

Introduction

- The **data link control** (**DLC**) deals with procedures for communication between two adjacent nodes (**node-to-node communication**).
- No matter whether the link is dedicated or broadcast.
- We discussed that data link control functions include framing and flow and error control.
- The other sublayer in data-link layer is MAC.

- DLC functions/services:
  - **Framing**
  - **Flow control**
  - **Error control**

简介

- 数据链路控制（**DLC**）处理两个相邻节点之间（**节点到节点通信**）的通信规程。
- 无论链路是专用的还是广播的。
- 我们讨论过，数据链路控制功能包括成帧以及流量和差错控制。

- 数据链路层中的另一个子层是MAC。

- DLC功能/服务：
  - **组帧**
  - **流量控制**
  - **差错控制**

Framing Analogy: Postal System

- The data-link layer needs to pack bits into **frames**, so that each frame is distinguishable from another.

类比说明：邮政系统

- 数据链路层需要将比特封装成**帧**，以便每个帧能够与其他帧区分开来。

Framing Analogy: Postal System

- The data-link layer needs to pack bits into **frames**, so that each frame is distinguishable from another.
- The message is usually packed into **multiple frames**, **why**?

框架类比：邮政系统

- 数据链路层需要将比特打包成**帧**，以便区分不同的帧。
- 消息通常被打包成**多个帧**，**为什么**?

Framing Analogy: Postal System

- The data-link layer needs to pack bits into **frames**, so that each frame is distinguishable from another.
- The message is usually packed into **multiple frames**, **why**?
  - Inefficient flow and error control if the frame is very large → a single-bit error requires retransmission of the whole frame

类比说明：邮政系统

- 数据链路层需要将比特封装成 **帧**，以便区分不同的帧。
- 消息通常被封装成 **多个帧**，**为什么**?
  如果帧非常大，则会导致效率低下的流量和错误控制：→ 单个比特错误就需要重传整个帧

Framing Analogy: Postal System

- The data-link layer needs to pack bits into **frames**, so that each frame is distinguishable from another.
- The message is usually packed into **multiple frames**, **why**?
  - Inefficient flow and error control if the frame is very large → a single-bit error requires retransmission of the whole frame
- Framing in the data-link layer separates a message from one source to a destination by adding a **sender address** and a **destination address**.

框架类比：邮政系统

- 数据链路层需要将比特封装成**帧**，以便区分不同的帧。
- 消息通常被封装成**多个帧**，**为什么**?
  - 如果帧非常大，则流量控制和错误控制效率低下：→ 单个比特错误就需要重传整个帧
- 数据链路层的成帧通过添加**发送方地址**和**目的地址**，将来自一个源站点发往目标站点的消息分离开来。

Frame Size

- **Frame size**
  - **Fixed** → no need to define boundaries, frame size is fixed!
  - **Variable** → prevalent in LANs (we need a way to define the end of a frame and the beginning of the next)

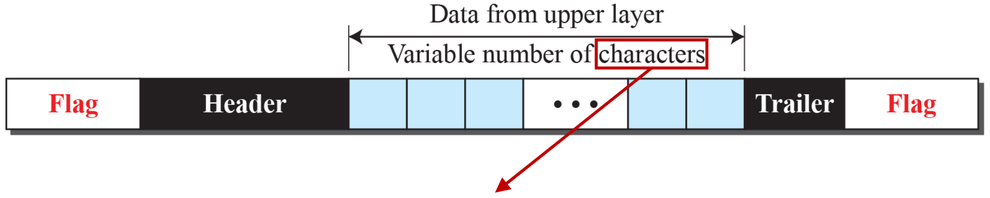
- To define the end of the frame we have two approaches:
  1. **Character (byte)-oriented approach**
  2. **Bit-oriented approach**

帧大小

- **帧大小**
  - **固定** → 无需定义边界 , 帧大小是固定的!
  - **可变** → 在局域网中很常见（我们需要一种方法来定义一帧的结束和下一帧的开始）

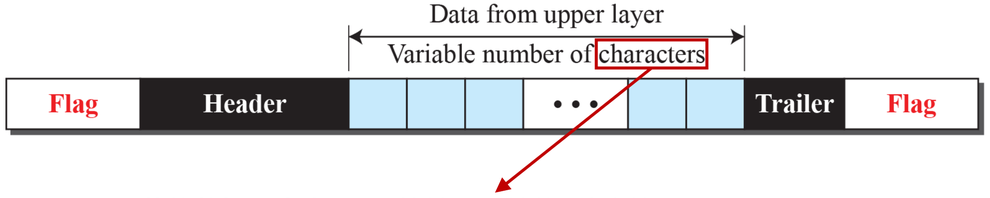
- 为了定义帧的结束，我们有两种方法：
  1. **面向字符（字节）的方法**
  2. **面向位的方法**

Character-Oriented Approach – Format of a Frame



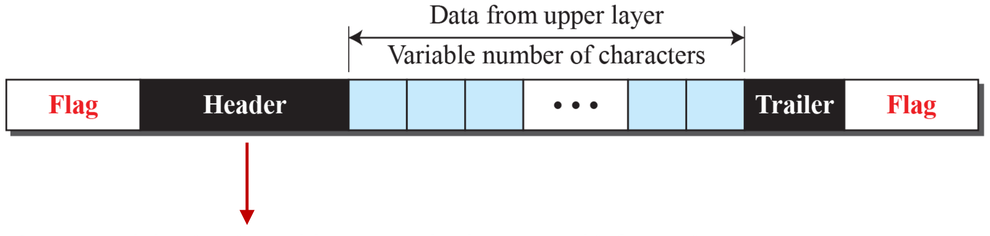
Data to be carried are 8-bit characters from a coding system such as ASCII.

面向字符的方法 — 一种帧



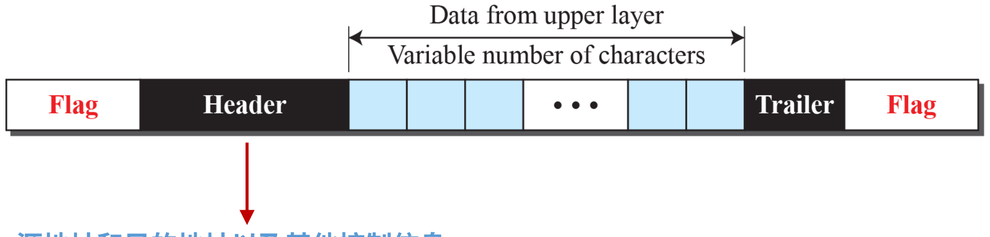
所承载的数据是来自ASCII等编码系统的8位字符。

Character-Oriented Approach – Format of a Frame



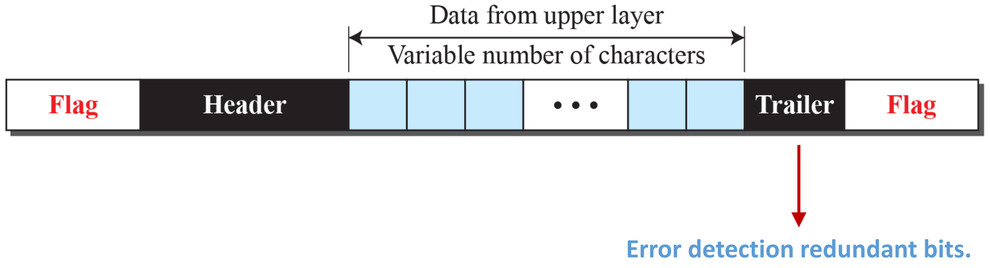
Source and destination addresses and other control information.

面向字符的方法——帧的格式

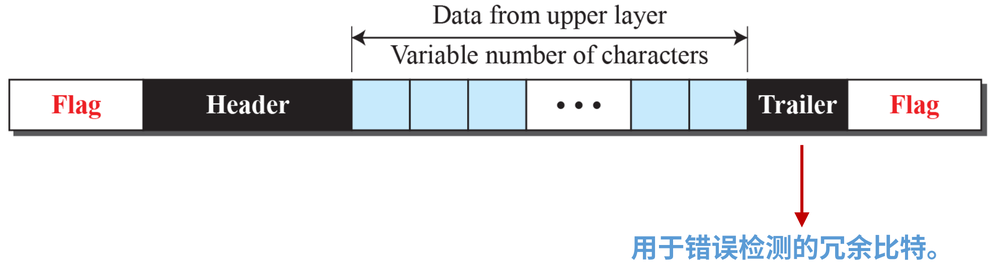


源地址和目的地址以及其他控制信息。

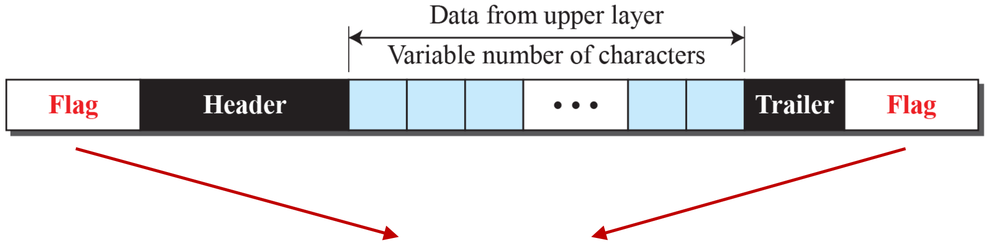
Character-Oriented Approach – Format of a Frame



面向字符的方法——帧的格式

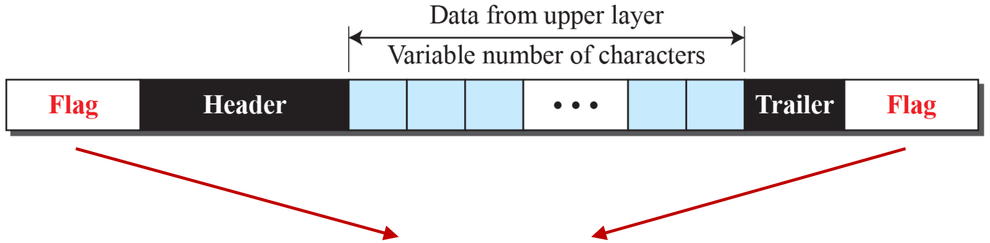


Character-Oriented Approach – Format of a Frame



To separate one frame from the next. It is composed of protocol-dependent special characters (8-bit (1-byte)).

面向字符的方法——帧的格式



用于将一个帧与下一个帧分隔开。它由协议相关的特殊字符（8位（1字节））组成。

Character-Oriented Approach – Problem

- Character-oriented framing was popular when only **text** was exchanged.
- In addition to text, we send other types of information such as graphs, audio, and video; **any character used for the flag could also be part of the information**. If this happens, the receiver, when it encounters this pattern in the middle of the data, **thinks** it has reached the end of the frame.

面向字符的方法——问题

- 当仅交换 **文本** 时，面向字符的组帧方式很流行。

- 除了文本外，我们还会发送图形、音频和视频等其他类型的信息；**任何用作标志的字符也可能是信息的一部分**。如果发生这种情况，接收方在数据中间遇到此模式时，**会认为** 已到达帧的末尾。

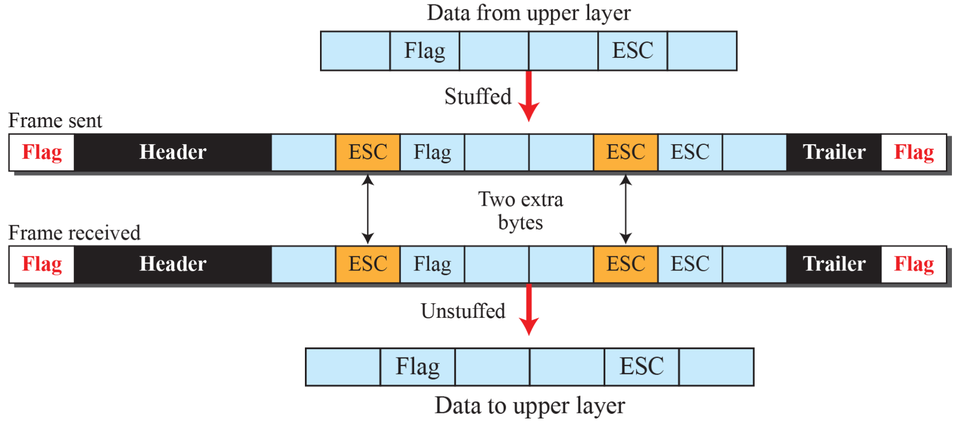
Character-Oriented Approach – Byte Stuffing and Unstuffing

- **Byte stuffing** (**character stuffing**) is the process of adding one extra byte (with a predefined bit pattern), which is called the escape character (ESC), whenever there is a flag or ESC in the text.
- Whenever the receiver encounters the **ESC character**, it removes it from the data section and treats the **next character** as **data**, not as a delimiting flag.

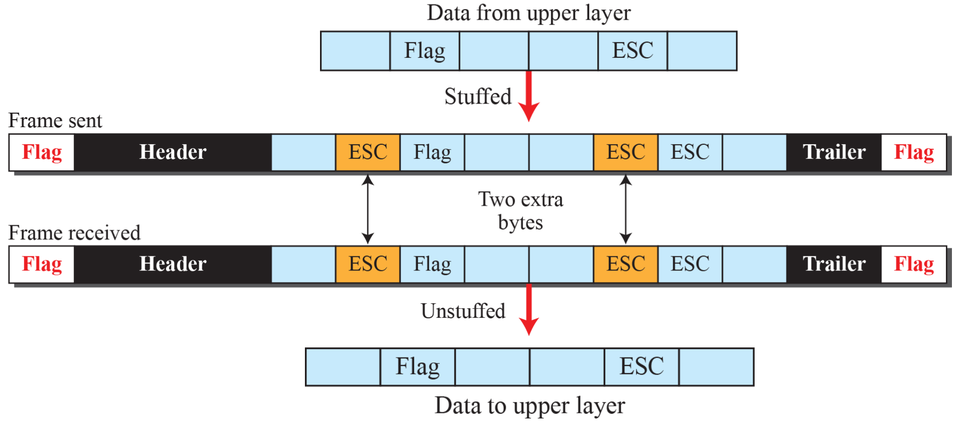
面向字符的方法——字节填充与解填充

- **字节填充** （**字符填充**）是指每当文本中出现标志或 ESC 字符时，添加一个具有预定义比特模式的额外字节，该字节称为转义字符（ESC）。
- 每当接收方遇到 **ESC 字符** 时，它会将其从数据部分删除，并将 **下一个字符** 视为 **数据**，而不是作为定界标志。

Byte Stuffing and Unstuffing



字节填充与去填充



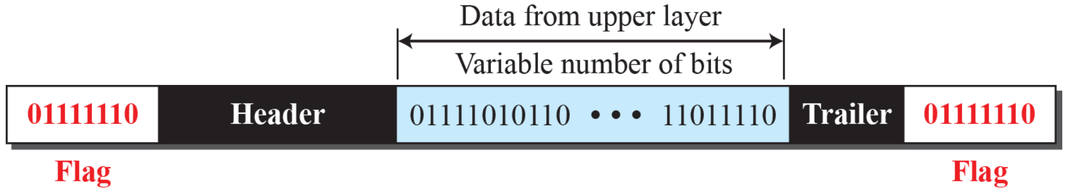
Character-Oriented Approach – The Other Problem!

- The universal coding systems in use today, such as Unicode, have 16-bit and 32-bit. Characters that conflict with 8-bit characters.
- In general, the tendency is moving toward the **bit-oriented** protocols.

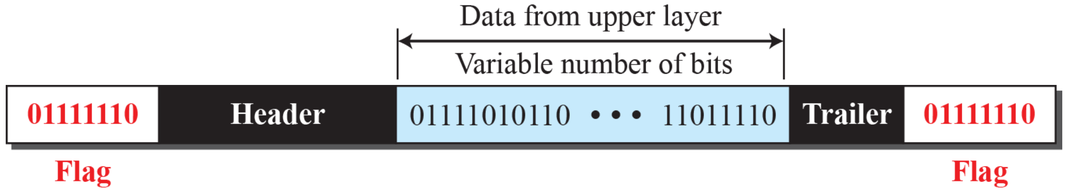
面向字符的方法——另一个问题！

- 当今使用的通用编码系统，例如 Unicode，采用 16 位和 32 位，其字符与 8 位字符发生冲突。
- 总体而言，趋势正朝着**面向比特**的协议发展。

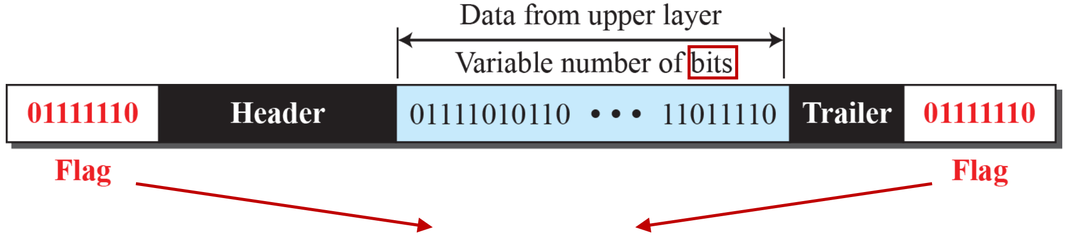
Bit-Oriented Approach



面向位的方法

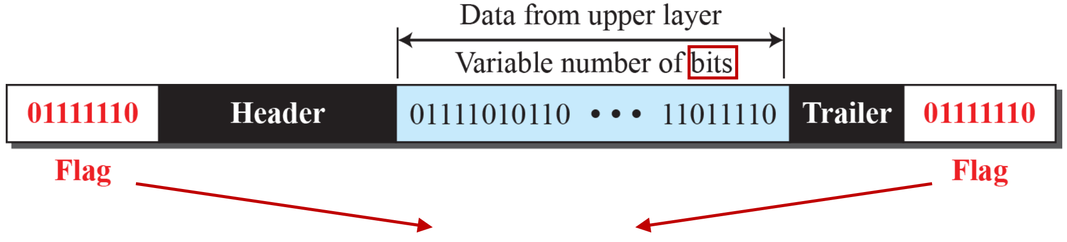


Bit-Oriented Approach



A special 8-bit pattern as the delimiter to define the beginning and the end of the frame – 01111110 is the common flag between most protocols.

面向比特的方法



使用一个特殊的8位模式作为分隔符，来定义帧的开始和结束——01111110 是大多数协议之间常用的标志。

Bit-Oriented Approach – Bit Stuffing and Unstuffing

- **Bit stuffing** is the process of adding one extra 0 whenever five consecutive 1s follow a 0 in the data, so that the receiver does not mistake the pattern **01111110** for a flag.

面向位的方法——位填充与位解填

- **位填充** 是指在数据中每当出现五个连续的1后跟一个0时，添加一个额外的0的过程，以防止接收方将 **01111110** 模式误认为标志位。

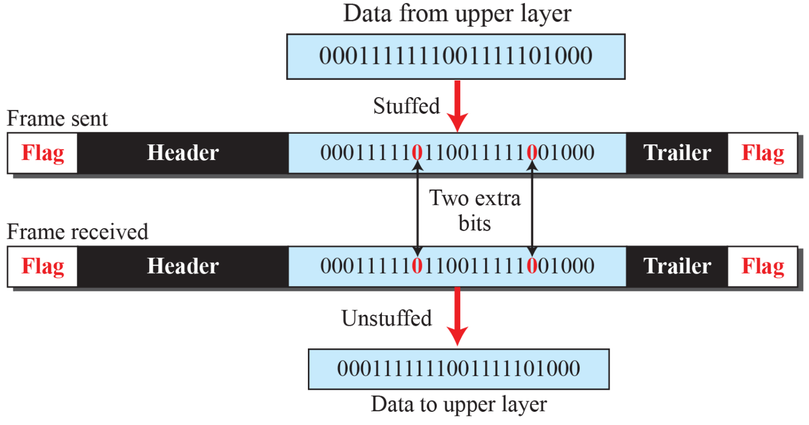
Bit-Oriented Approach – Bit Stuffing and Unstuffing

- **Bit stuffing** is the process of adding one extra 0 whenever five consecutive 1s follow a 0 in the data, so that the receiver does not mistake the pattern **01111110** for a flag.

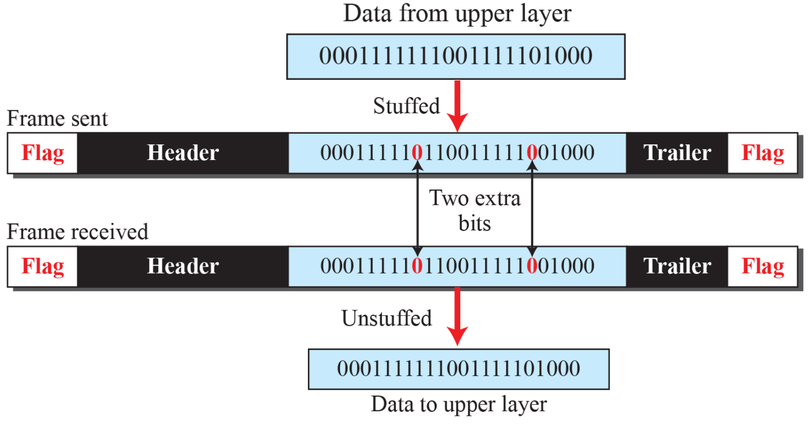
面向比特的方法——比特填充与解填充

- **比特填充** 是指每当数据中出现五个连续的1后跟一个0时，就添加一个额外的0的过程，以防止接收方将 **01111110** 这种模式误认为是标志位。

Bit-Oriented Approach – Bit Stuffing and Unstuffing



面向比特的方法——比特填充与解填充



Flow Control

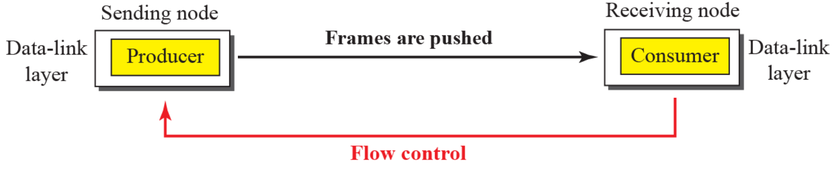
- Whenever an entity produces items and another entity consumes them, there should be a **balance** between **production** and **consumption rates**.
- If the **rate** of **produced** frames is higher than the **rate** of **consumed** frames, frames at the **receiving end** need to be **buffered** while waiting to be consumed (processed).
- A **buffer** is a set of memory locations that can hold packets at the sender and receiver.

流控制

- 每当一个实体生成项目而另一个实体消耗这些项目时，**生成**与**消耗速率**之间**应保持**平衡。
- 如果**生成**帧的**速率**高于**消耗**帧的**速率**，则接收端的帧在等待被处理（消耗）时需要被**缓冲**。
- 缓冲区**是一组内存位置，可用于在发送方和接收方存储数据包。**

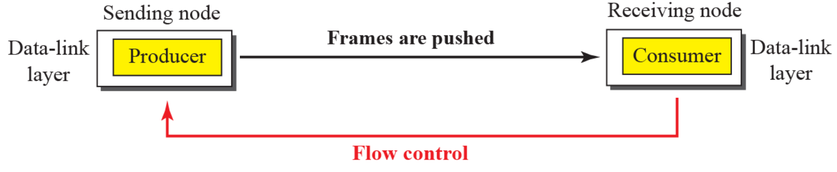
Flow Control (Cont.)

- Different strategies to implement flow control
  1. The receiving data-link layer **drops the frames** if its buffer is full.
  2. Receiving data-link layer sends a **feedback** to the sending data-link layer to ask it to stop or slow down.



流控制（续）

- 实现流控制的不同策略
  1. 接收端数据链路层**缓冲区满时丢弃帧**。
  2. 接收端数据链路层向发送端数据链路层发送**反馈**，要求其停止或减慢发送速度。



Error Control

- A CRC is added to the frame header by the sender and checked by the receiver.
- Two approaches
  1. Corrupted frames are **discarded**, and the uncorrupted ones are delivered to the network layer.
     - Mostly used in wired LANs such as Ethernet
  2. Corrupted frames are **discarded**, and an **acknowledgment** is sent (for the purpose of both flow and error control) to the sender for the **uncorrupted frames**.

错误控制

- 发送方在帧头添加一个CRC，由接收器
- 两种方法
  1. 损坏的帧被**丢弃**，未损坏的帧则递交给网络层。
     - 主要用在以太网等有线局域网中
  2. 损坏的帧被**丢弃**，并且向发送方发送一个**确认**（用于流量和差错控制）以确认**未损坏的帧**。

Combination of Flow and Error Control

- The **acknowledgment** that is sent for **flow control** can also be used for **error control** to tell the sender the packet has arrived **uncorrupted**.
- The lack of acknowledgment means that there is a problem in the sent frame.

流量控制与差错控制的结合

- 用于**流量控制**而发送的**确认**也可用于**差错控制**，以告知发送方数据包已到达**且未受损**。

缺少确认意味着所发送的帧存在问题。

Data-Link Layer Protocols

- Two most commonly used DLC protocols to deal with flow and error control
  1. **Simple**
  2. **Stop-and-Wait**

- The behavior of a data-link-layer protocol can be better shown as a finite state machine (FSM).

数据链路层协议

- 两种最常使用的数据链路控制（DLC）协议用于处理流量和错误控制
  1. **简单**
  2. **停等协议**

- 数据链路层协议的行为可以用有限状态机（FSM）更清晰地表示。

Finite State Machines (FSMs)

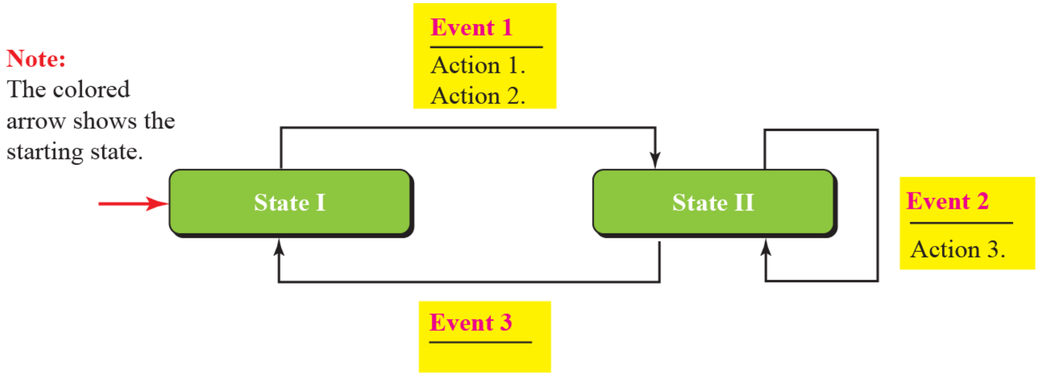
- FSM is used to simulate sequential logic
- An FSM is thought of as a machine (system) with a finite number of states.
- The machine is always in one of the states until an event occurs.
  - **Rounded-corner rectangles** or **circles** to show **states**.
  - **Colored text** to show **events**.
  - **Regular black text** to show **actions**.

有限状态机（FSM）

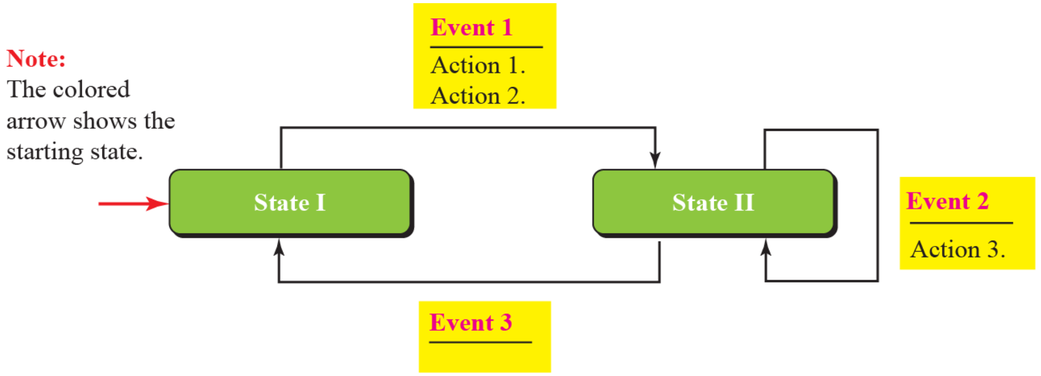
- FSM 用于模拟时序逻辑
- 有限状态机被视为具有有限数量状态的机器（系统）。

- 该机器始终处于其中一个状态，直到发生事件。
  - **圆角矩形** 或 **圆形** 用于表示 **状态**。
  - **彩色文本** 用于表示 **事件**。
  - **常规黑色文本**用于显示**操作**。

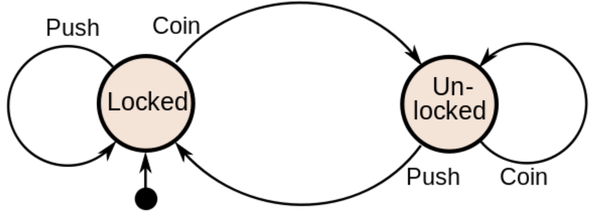
Finite State Machines (FSMs)



有限状态机（FSMs）

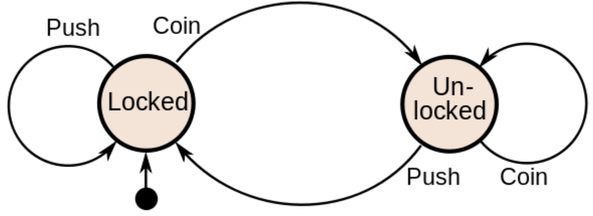


Finite State Machines (FSMs) – Example



Current State	Input	Next State	Output
Locked	coin	Unlocked	Unlocks the turnstile so that the customer can push through.
	push	Locked	None
Unlocked	coin	Unlocked	None
	push	Locked	When the customer has pushed through, locks the turnstile.

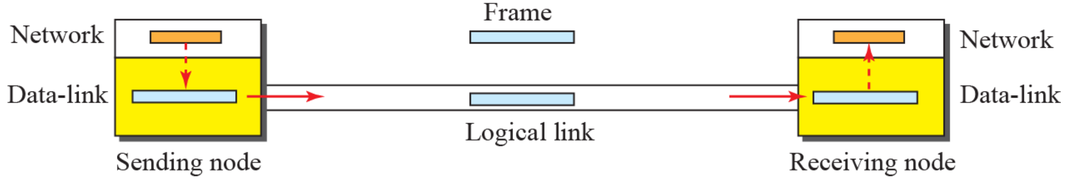
有限状态机（FSM）—— 示例



Current State	Input	Next State	Output
Locked	coin	Unlocked	Unlocks the turnstile so that the customer can push through.
	push	Locked	None
Unlocked	coin	Unlocked	None
	push	Locked	When the customer has pushed through, locks the turnstile.

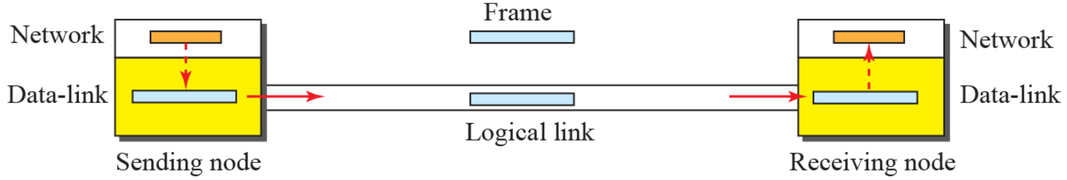
Simple Protocol

- **No flow and error control**.
- Assumption is that the receiver can never be overwhelmed with incoming frames.

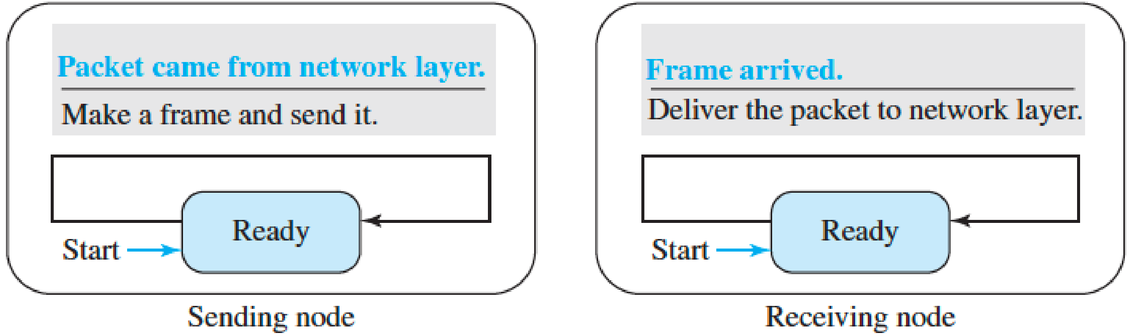


简单协议

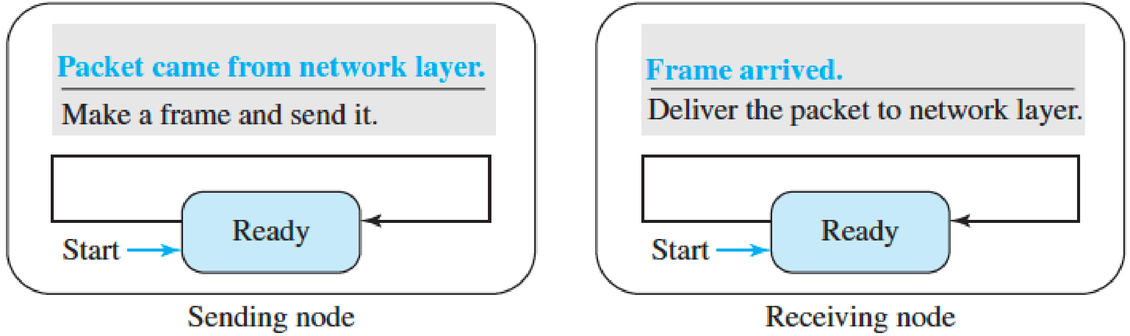
- **无流量控制和差错控制**。
- 假设接收方永远不会因传入的帧。



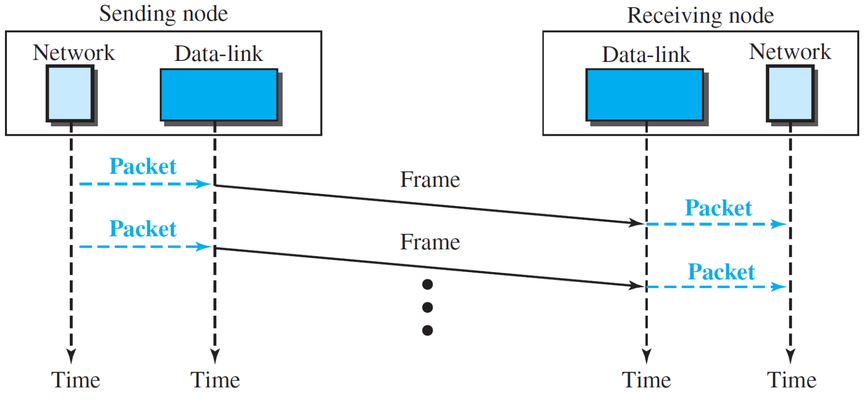
Simple Protocol – FSM



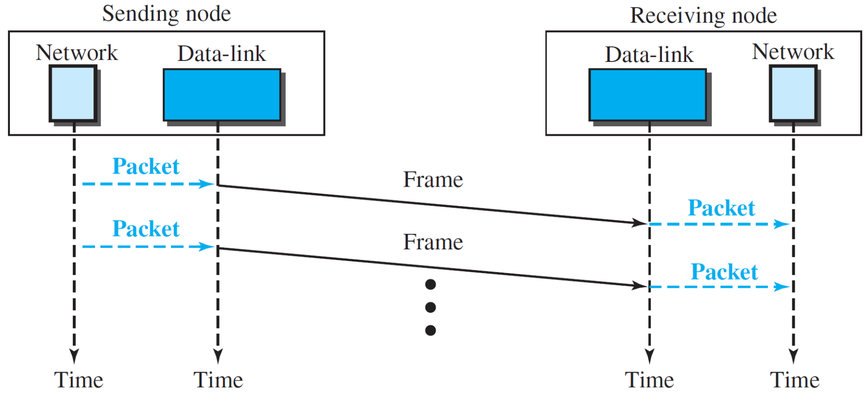
简单协议——有限状态机



Simple Protocol – Flow Diagram

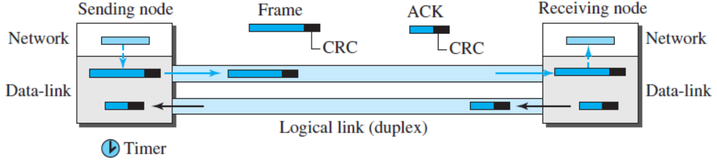


简单协议——流程图



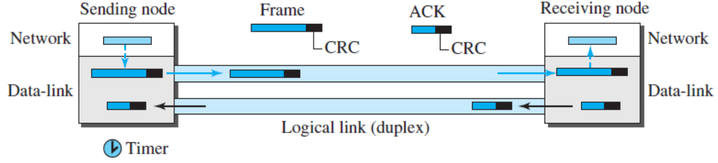
Stop-and-Wait Protocol

- A **connection-based** (connection-oriented) protocol
- **Both flow and error control provided**.
- The sender sends one frame at a time and waits for an acknowledgment before sending the next one.
- A **timer** used by the sender (if the timer expires, the frame is retransmitted).
- If the CRC is correct, the receiver will send an acknowledgement.

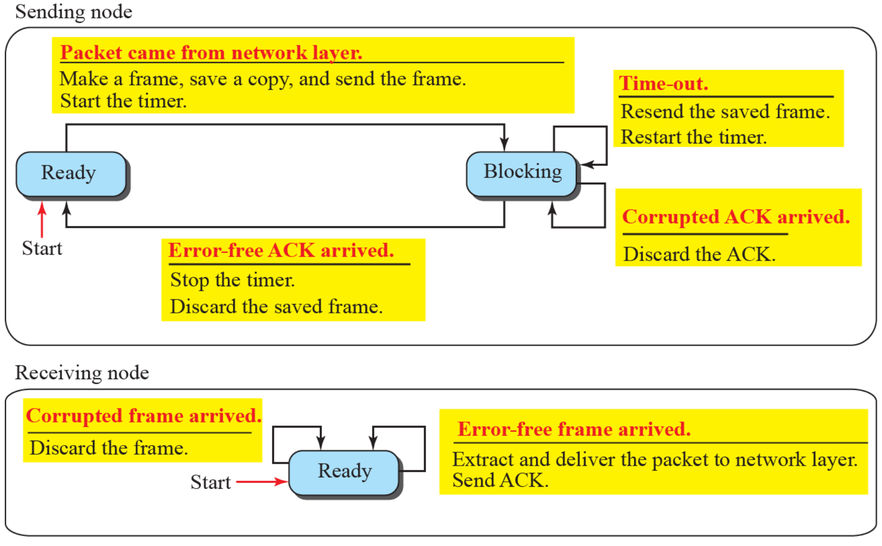


停止-等待协议

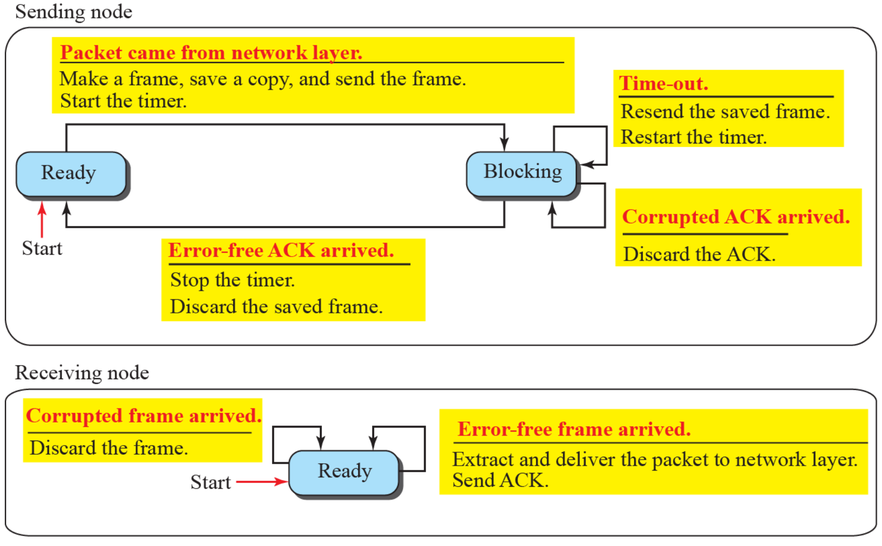
- 一种 **基于连接**（面向连接）的协议
- **提供流量控制和差错控制**。
- 发送方一次发送一个帧，并在发送下一个帧之前等待确认。
- 发送方使用一个 **定时器** （如果定时器超时，则重传该帧）。
- 如果CRC正确，接收方将发送一个确认。



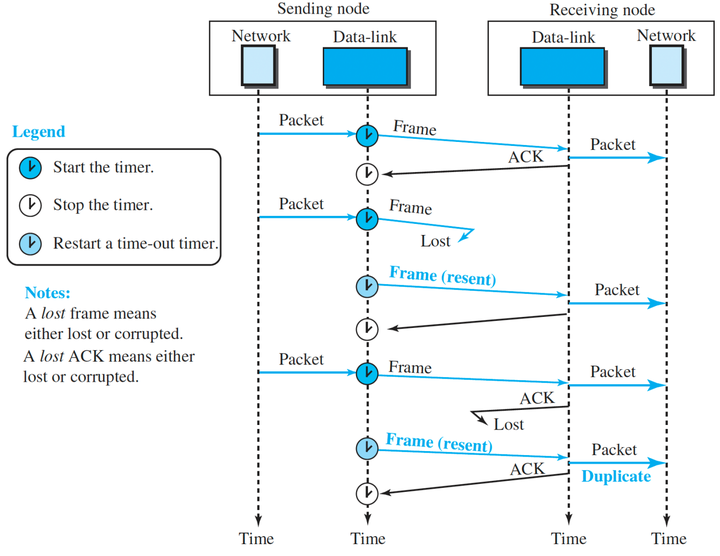
Stop-and-Wait Protocol – FSM



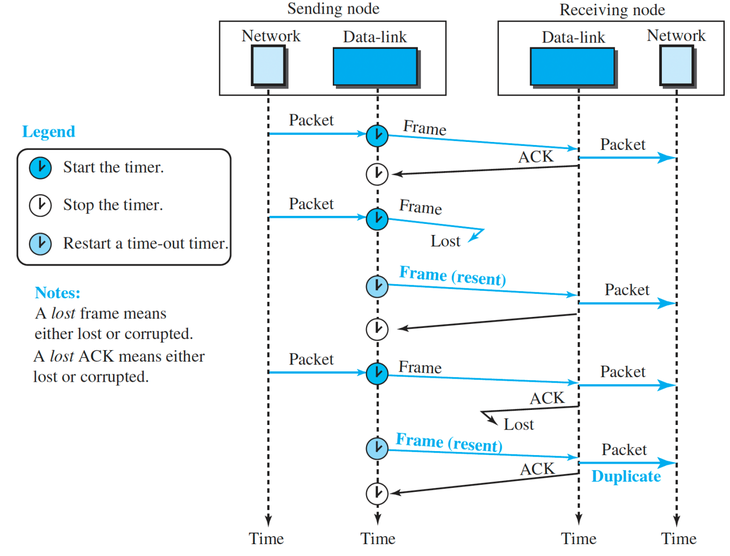
停止等待协议——有限状态机



Stop-and-Wait Protocol – Flow Diagram



停止等待协议——流程图



Stop-and-Wait Protocol

- **Duplicates** and **orders** can be handled by adding a **sequence number** and **acknowledgment number** to the frames.
  - Sequence numbers are 0, 1, 0, 1, 0, 1, . . .
  - Acknowledgment numbers are 1, 0, 1, 0, 1, 0, …
- In other words, the sequence numbers start with 0, the acknowledgment numbers start with 1.
- An acknowledgment number always define the sequence number of the next frame to receive.
- A similar concept in TCP in transport layer → We'll discuss later in the course

停止-等待协议

- **重复** 和 **顺序** 可通过在帧中添加 **序列号** 和 **确认号** 来处理。
  - 序列号为 0、1、0、1、0、1，……
  - 确认号为 1、0、1、0、1、0，……
- 换句话说，序列号从 0 开始，确认号从 1 开始。
- 确认号始终定义下一个要接收的帧的序列号。
- 传输层 TCP 中的一个类似概念 → 我们将在课程的后续部分讨论

Summary

- DLC responsibilities/services
  - Framing → Variable-size frames
  - Flow control
  - Error control
- DLC Protocols
  - Simple protocol
  - Stop-and-wait protocol

摘要

- 数据链路控制职责/服务
  - 组帧 → 可变尺寸帧
  - 流量控制
  - 差错控制
- 数据链路控制协议
  - 简单协议
  - 停止-等待协议

References

[1] Behrouz A.Forouzan, Data Communications & Networking with TCP/IP Protocol Suite, 6th Ed, 2022, McGraw-Hill companies.

参考文献

[1] Behrouz A.Forouzan, Data Communications & Networking with TCP/IP 协议套件，第6版，2022年，麦格劳-希尔公司。

Reading

- Chapter 3 of the textbook, sections 3.2.1 and 3.2.3.
- Chapter 3 of the textbook, section 3.6 (Practice Test)

阅读

- 教材第3章，第3.2.1节和第3.2.3节。
- 教材第3章，第3.6节（练习测试）