Data Communication
(DC)

Lecture 4a

Overview of the contents

- **Data Link Control Services**
  - **Framing**
  - **Flow and error control**
- **Data Link Layer Protocols**
  - **Simple protocol**
  - **Stop-and-Wait protocol**
  - **Piggybacking**

Data Link layer

Two main functions (sublayers) of the Data Link layer are:

- **Data Link control**. communication between two neighboring nodes is handled via design and procedures. i.e., node-to-node communication. It deal with all issues common to both point-to-point and broadcast links.

- **Media Acess Control (MAC)**. Here discuss how a link can be shared between several nodes, and how to control that to takes place, i.e., multi-node communication. It deals only with issues specific to broadcast links.

Data Link layer

Data **L**ink **C**ontrol (**DLC**)

Data Link control functions include:

- Framing
- Flow control
- Error control

There are software-implemented protocols that offer these features and provide a smooth and reliable transmission.

Regarding flow and error control, We have already looked at some methods for detecting and/or correcting errors (Hamming distance, CRC, etc.).

Data Link layer

Data **L**ink **C**ontrol (**DLC**)

In order to implement Data Link control, we need to have some protocols, i.e., a set of rules that both sender and receiver comply with.

We will look at some of these protocols:

- Simple protocol
- Stop-and-Wait protocol
- Stop-and-Wait protocol with numbering

Finally, we will look at how a bit-oriented protocol is actually implemented.

For this purpose, we use
High-level Data Link Control (HDLC) protocol.
We are also looking at another popular (byte-oriented) protocol, namely
Point-to-Point Protocol (PPP).

Data Link layer

Framing

In the Data Link layer, we pack the bit stream into what we call **Frames**, so that the individual frames can be distinguished from each other.

The division of frames can take place in:

- **Fixed sizes** (e.g., ATM Wide Area Network).
- **Variable sizes** (most of LAN Networks)

We will concentrate on the variable-size framing, which is prevalent in local-area networks.

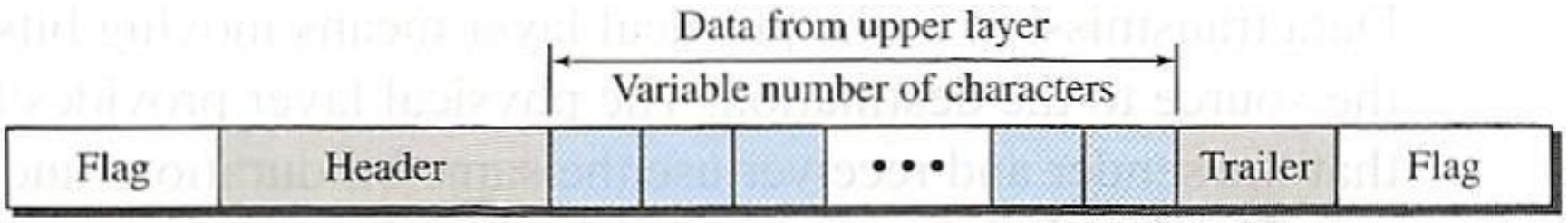
Historically, there are two variable-size framing methods here:

- **Character-oriented**
- **Bit-oriented**

But common to both methods is that there must be a **start-** and **end-flag**

Data Link layer

Character-oriented framing



Here the information is carried by 8 bits as a unit (one byte e.g., ASCII).
A special character (which does not appear in the data stream) is used as the **flag**

- **The header contains:** source and destination address and other control information.

- **The trailer contains:** the redundant bits used to detect errors (and perhaps correct them).

Data Link layer

Character-oriented framing

But as the information could also be graphics, audio and video. Then it could no longer be guaranteed that the flag could not occur in the data stream itself.

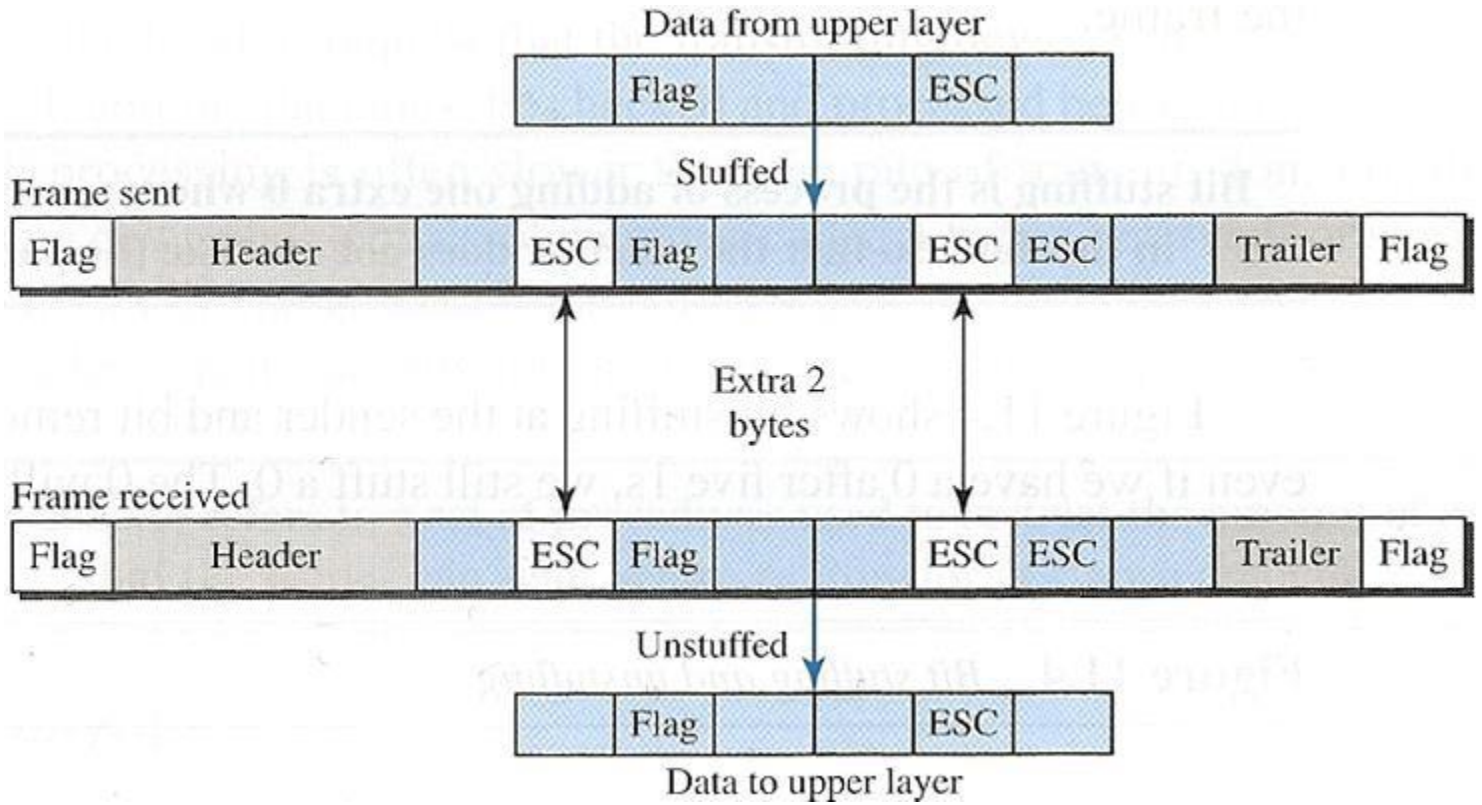
This would be mistaken for the end of a Frame.

Data Link layer

Character-oriented framing: Byte stuffing

Here a special character (Esc code) is used, which is added in the data section just before the characters that have the same pattern as the flag.

However, it may also happen that the Esc code itself appears in the data section. If this happens, an Esc code is inserted here just before this character.



Data Link layer

Character-oriented framing

Today, character-oriented protocols have another problem in data communication, as 16- and 32-bit universal coding systems are more popular, such as UniCode. This conflicts with the old-fashioned 8-bit characters.

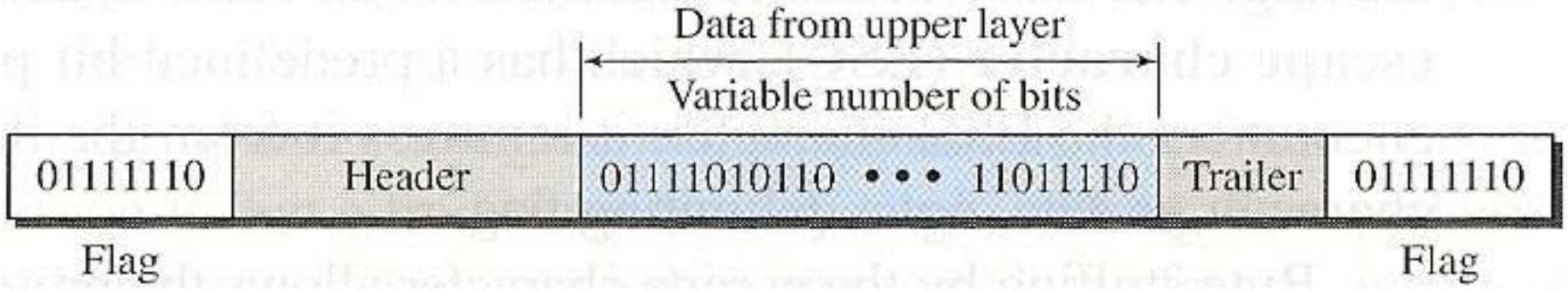
We can generally say that the trend is moving away from the character-oriented framing and in the direction of the bit-oriented framing.

Data Link layer

Bit-oriented framing

In this type of framing, the data section consists of a sequence of bits. These are then interpreted by the upper layers as: text, graphics, audio and video, respectively.
Besides Header and (maybe) Trailer, we still need a flag to delimit the individual Frames.

Most protocols use an 8-bit pattern **01111110** as the flag.

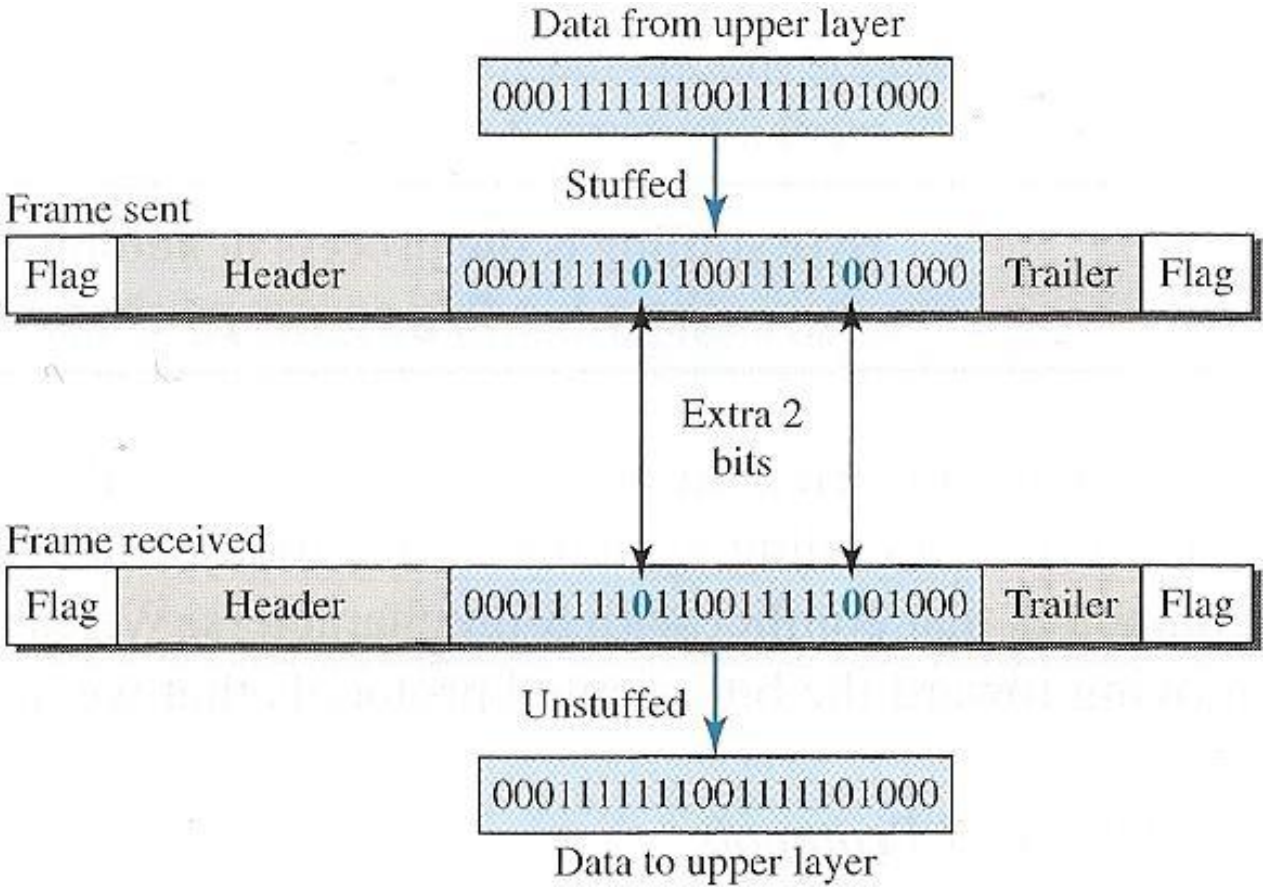


Note the same issue with the flag pattern appearing in the data stream

Data Link layer

Bit-oriented framing: Bit stuffing

Method:
If a **0** and 5 consecutive **1s** exist, then an extra 0 is added right after them regardless of which bit (0 or 1) coming after this sequence.



Data Link layer

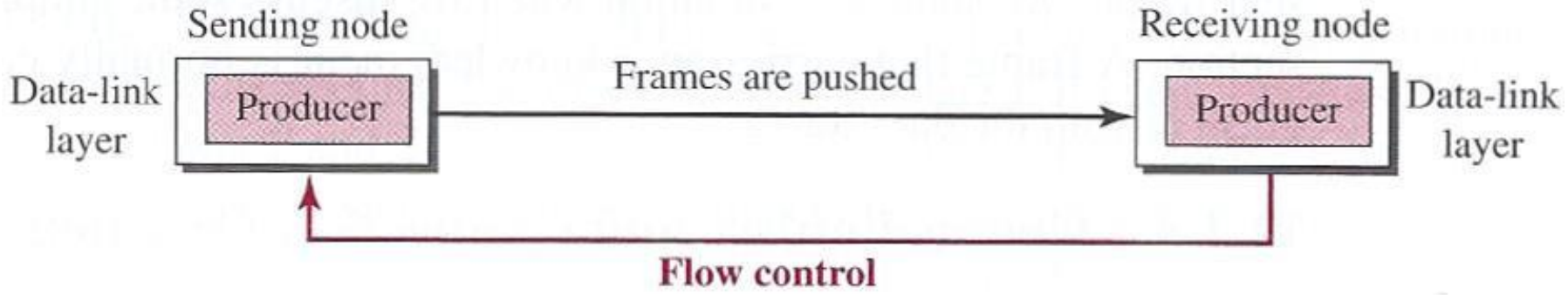
Flow and error control

Flow control coordinates the amount of data that can be sent before an acknowledgment is to be received.

This is one of the most important tasks for the Data Link layer.

Every receiving node has a limit to the speed at which they can receive data. E.g.:

- Limited memory for storing incoming data (buffers).
- The time it takes to process incoming data.



Data Link layer

Flow and error control

Error control. This involves both error detection and correction. This control allows the receiver to inform the sender of the lost or corrupted frames during transmission and to coordinate a retransmission of these frames.

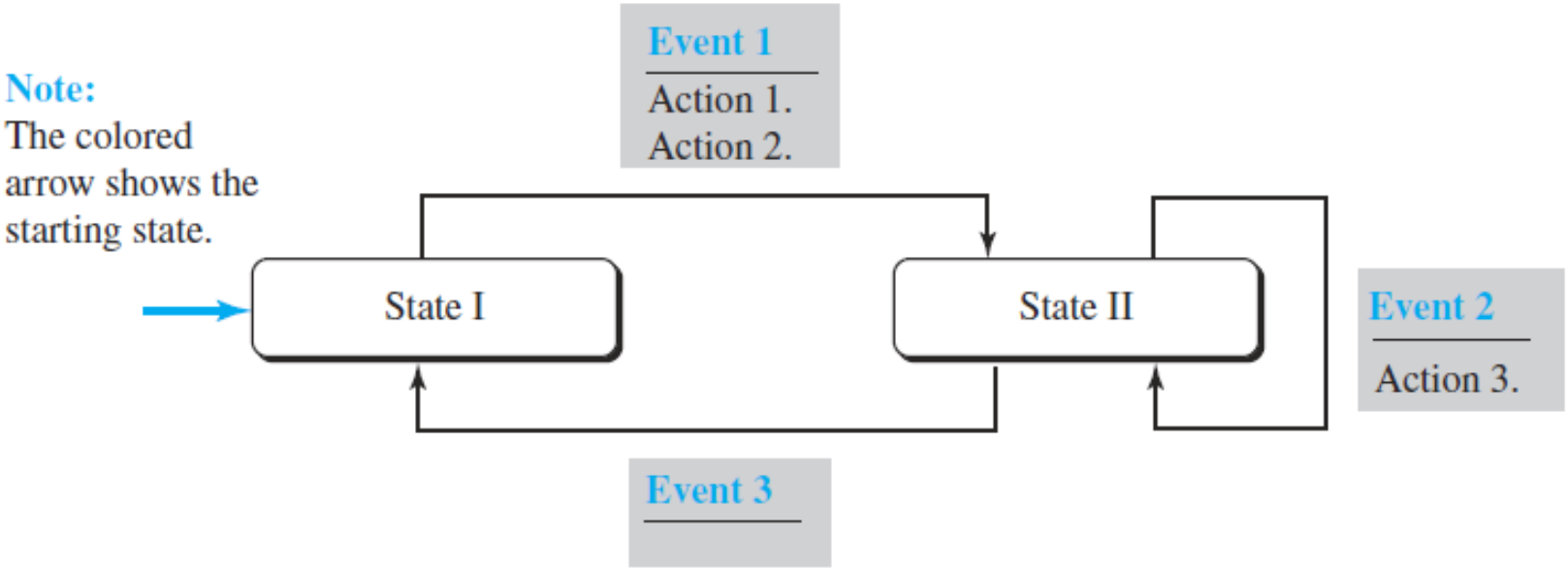
In the Data Link layer, the primary method of error control is Retransmission.

This method is called **Automatic Repeat reQuest (ARQ)**.

Data Link layer

Finite State Machine (FSM)

An FSM is a machine with a finite number of states. The machine is always in one of the states until an event occurs. Each event is associated with two reactions: **defining the list (possibly empty) of actions to be performed** and **determining the next state (which can be the same as the current state)**. One of the states must be defined as the initial state, the state in which the machine starts when it turns on.

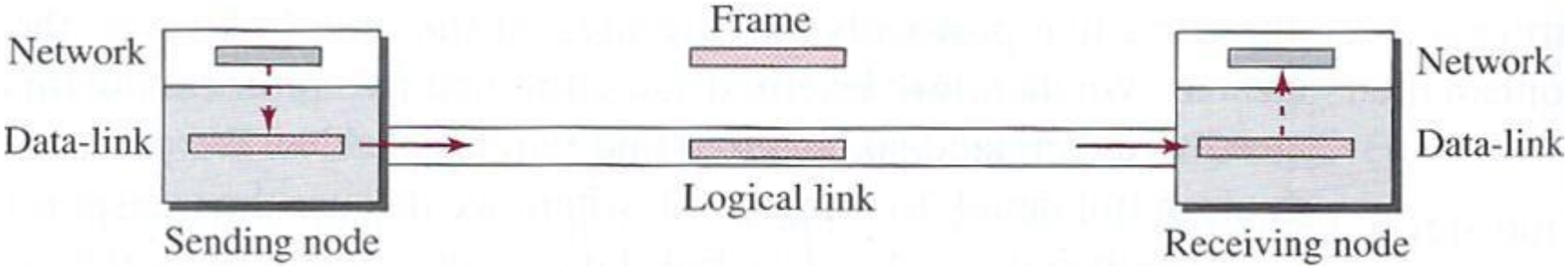


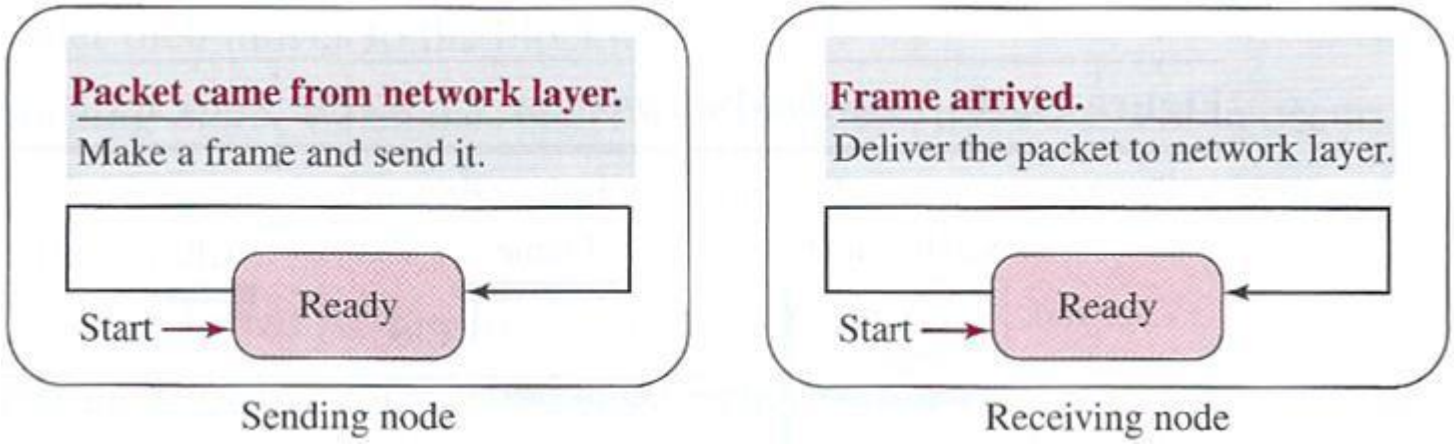
Data Link layer

Protocols: Simple protocol

It is assumed that the receiver can immediately handle any frame it receives.

In other words, the receiver can never be overwhelmed with incoming frames.

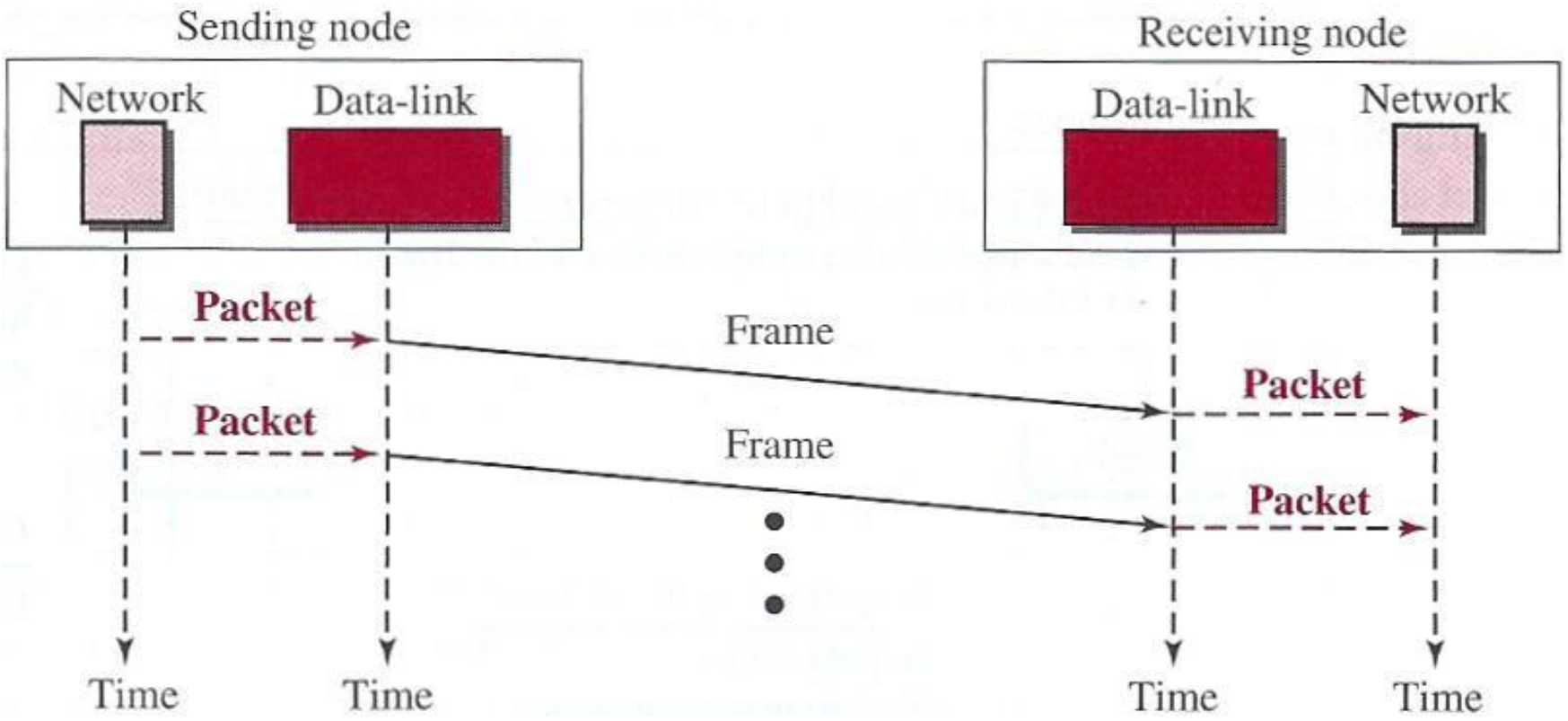




Note: simple protocol has neither flow control or error control

Data Link layer

Protocols: Simple protocol - flow diagram



Notice that you can see the transmission times

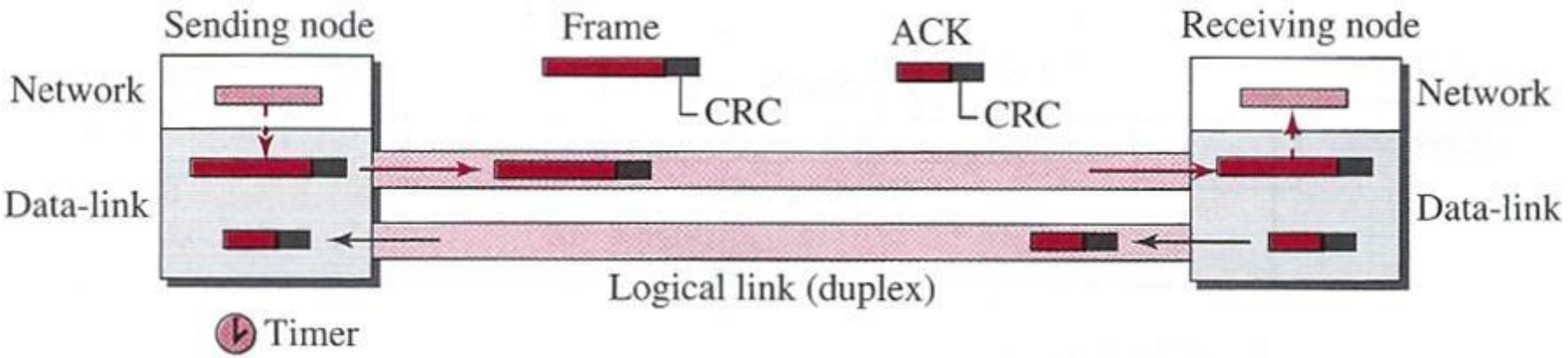
Data Link layer

Protocols: Stop-and-Wait protocol

In the Stop-and-wait protocol, only one frame is sent at a time, then an acknowledge frame (ACK) is awaited before the next frame is sent.

A CRC is used for each data frame and ACK frame to detect errors.

A time at sender side is used for ARQ.

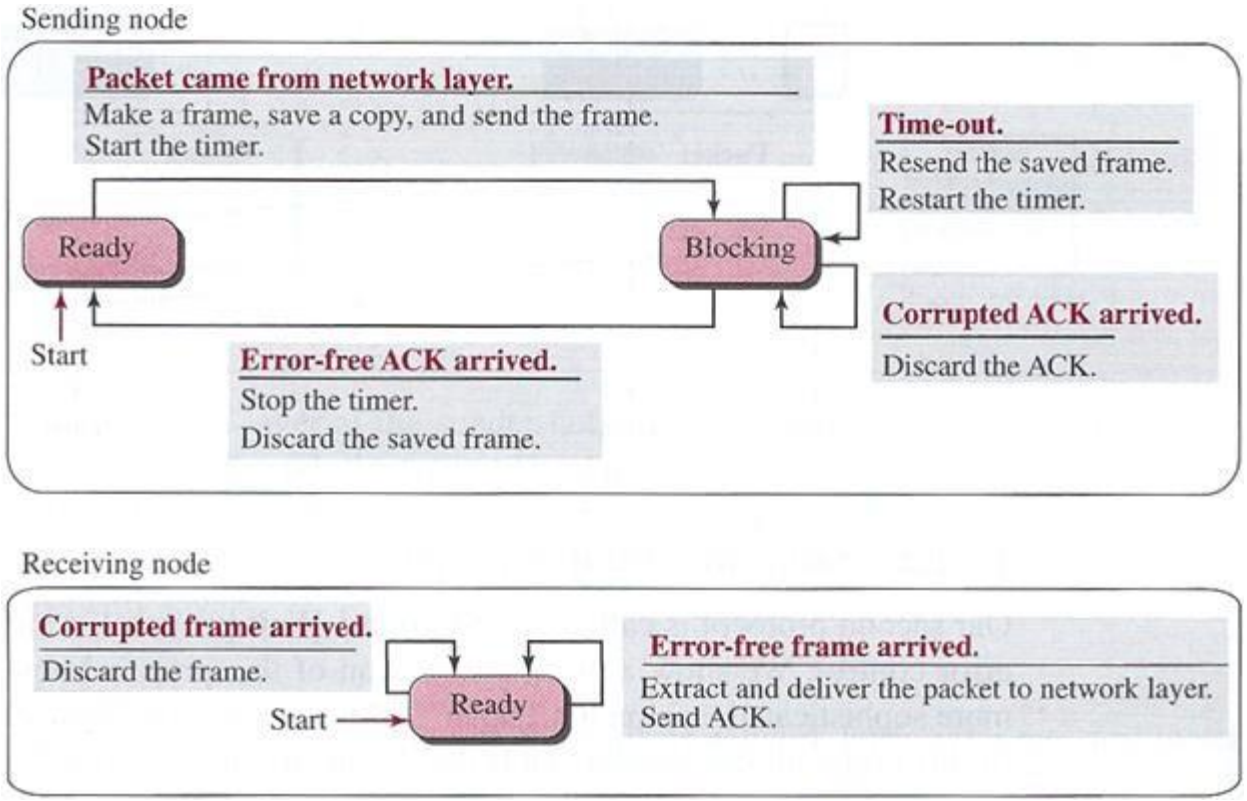


Note that only one frame and one acknowledgment can be in the channels at any time.

Data Link layer

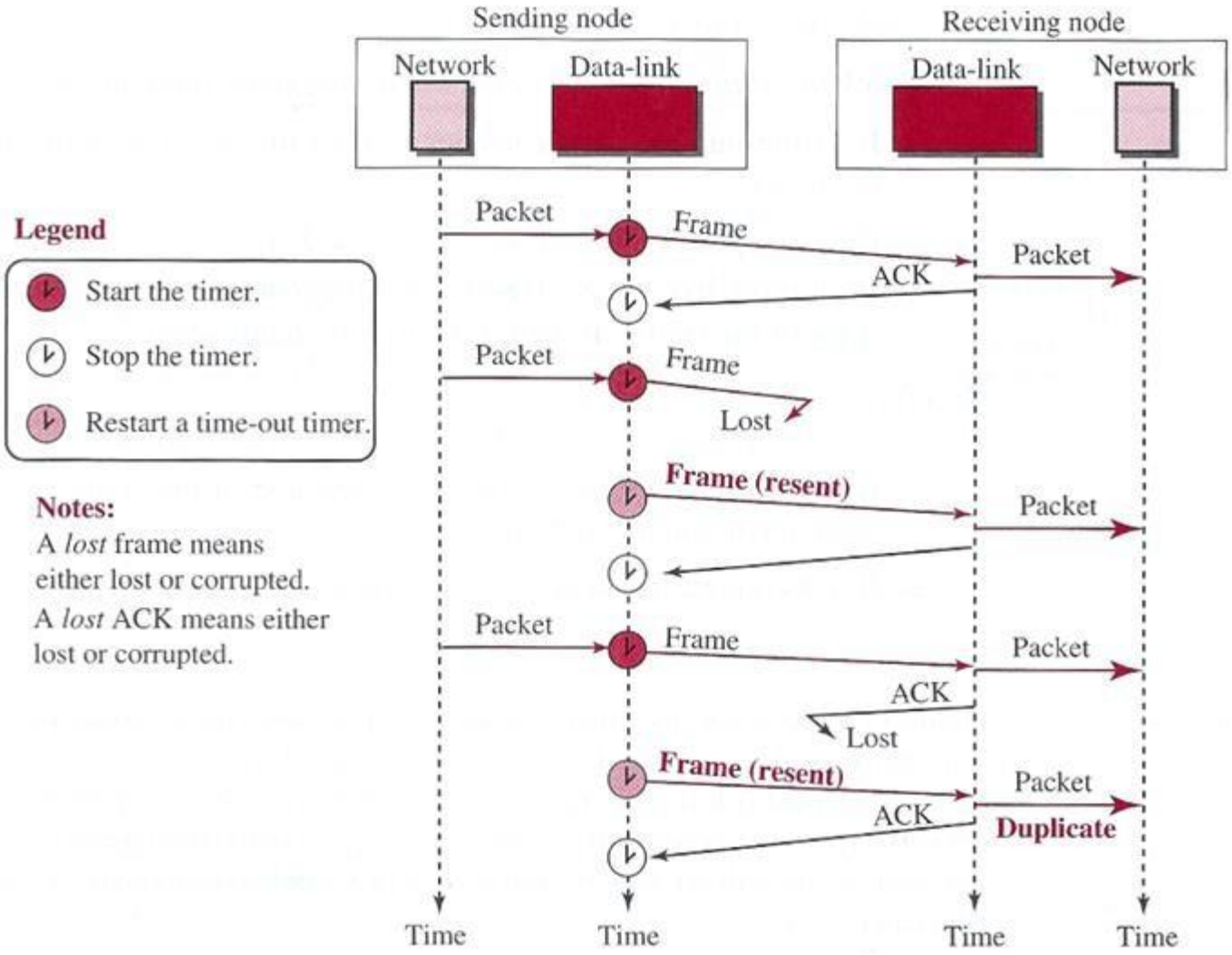
Protocols: Stop-and-Wait protocol

When a frame is received, it is checked for errors, if it is corrupted it is simply discarded. When an error is found, cf. this protocol, this is manifested by the receiver remaining silent.



Data Link layer

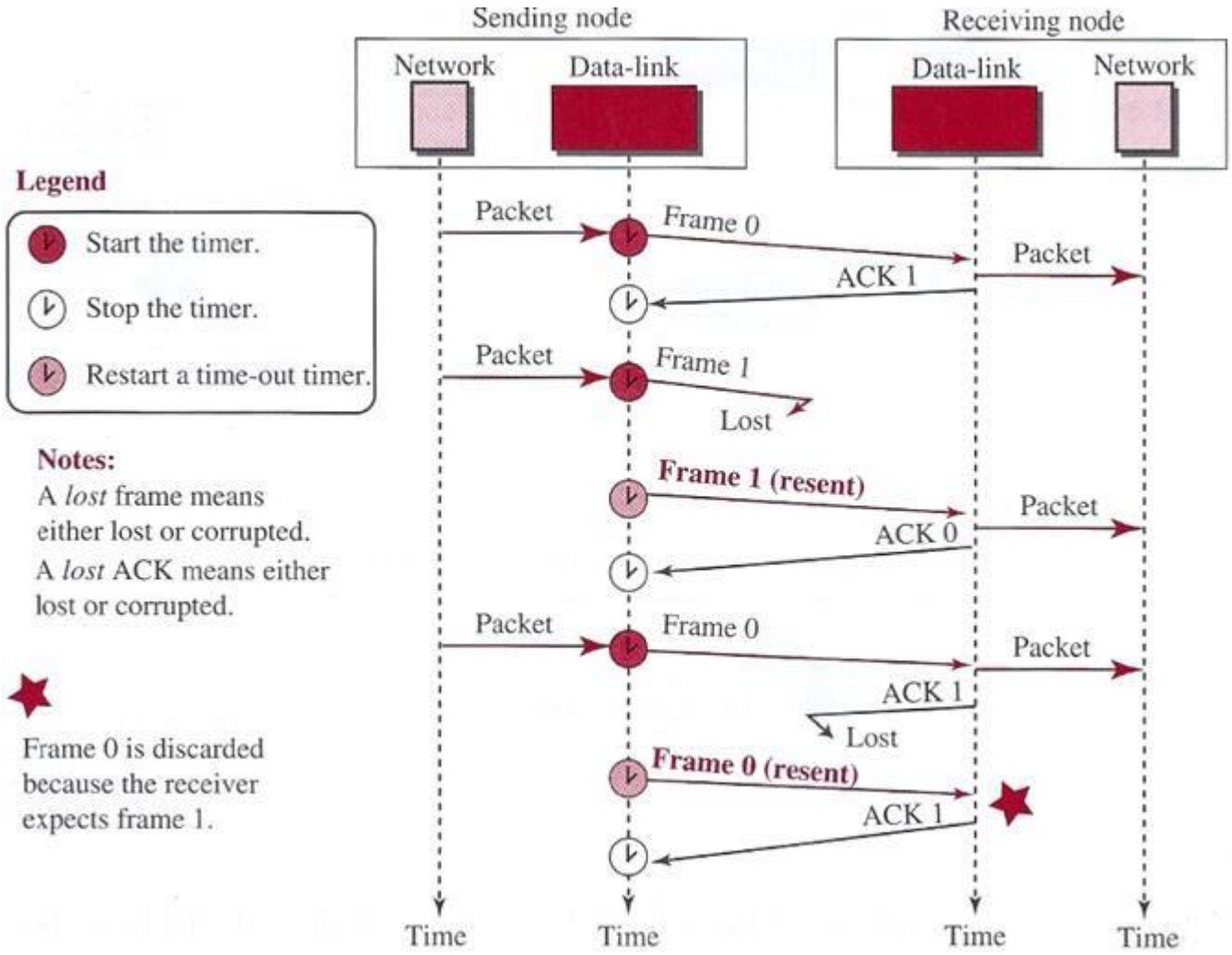
Protocols: Stop-and-Wait protocol - flow diagram



NOTE that this protocol cannot deal with duplicates! If an ACK frame is lost

Data Link layer

Protocols: Stop-and-Wait protocol with numbering



NOTE: duplicate frames can be caught because of incorrect numbers

Data Link layer

Protocols: Stop-and-Wait protocol with numbering

Sequence numbering

An important consideration here is how many different sequence numbers we need to have in order to make a unique communication.

if a sequence number has ***m*** bits, then the sequence numbers can go from ***0*** to 2^m-1, then they are repeated.

In this protocol only 2 different sequence numbers are needed since one acknowledgment frame (ACK) for each Frame before the next is sent.

An ACK Frame sent as a receipt for a correctly received frame contains the sequence number of the next frame that the receiver expects to receive.

Data Link layer

Example of Piggybacking