Chapter 3

The Data Link Layer

Data Link Layer Design Issues
- Services Provided to the Network Layer
- Framing
- Error Control
- Flow Control

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

Functions of the Data Link Layer (2)

Relationship between packets and frames.
Services Provided to Network Layer

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

Services Provided to Network Layer (2)

- Placement of the data link protocol.

Framing

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

Framing (2)

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

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

Error Detection and Correction

- Error-Correcting Codes
- Error-Detecting Codes

Error-Correcting Codes

<table>
<thead>
<tr>
<th>Char.</th>
<th>ASCII</th>
<th>Check bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>100100</td>
<td>001100010000</td>
</tr>
<tr>
<td>a</td>
<td>110001</td>
<td>01110010001</td>
</tr>
<tr>
<td>m</td>
<td>1101011</td>
<td>101100101010101</td>
</tr>
<tr>
<td>i</td>
<td>110101</td>
<td>110101010101</td>
</tr>
<tr>
<td>n</td>
<td>1101101</td>
<td>010101010001</td>
</tr>
<tr>
<td>g</td>
<td>1100111</td>
<td>011100111111</td>
</tr>
<tr>
<td>t</td>
<td>0100000</td>
<td>100110000000</td>
</tr>
<tr>
<td>c</td>
<td>1100011</td>
<td>111100000111</td>
</tr>
<tr>
<td>o</td>
<td>1101111</td>
<td>101010111111</td>
</tr>
<tr>
<td>d</td>
<td>1100100</td>
<td>11110011000</td>
</tr>
<tr>
<td>e</td>
<td>110101</td>
<td>00110001011</td>
</tr>
</tbody>
</table>

Order of bit transmission

Use of a Hamming code to correct burst errors.

Error-Detecting Codes

Calculation of the polynomial code checksum.
Elementary Data Link Protocols

- An Unrestricted Simplex Protocol
- A Simplex Stop-and-Wait Protocol
- A Simplex Protocol for a Noisy Channel

Sliding Window Protocols

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

Sliding Window Protocols (2)

A sliding window of size 1, with a 3-bit sequence number.
(a) Initially.
(b) After the first frame has been sent.
(c) After the first frame has been received.
(d) After the first acknowledgement has been received.

A Protocol Using Go Back N

Pipelining and error recovery. Effect on an error when
(a) Receiver’s window size is 1.
(b) Receiver’s window size is large.
A Sliding Window Protocol Using Selective Repeat (5)

Sender: 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
Receiver: 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7

(a) Initial situation with a window size seven.
(b) After seven frames sent and received, but not acknowledged.
(c) Initial situation with a window size of four.
(d) After four frames sent and received, but not acknowledged.

Protocol Verification

• Finite State Machine Models
• Petri Net Models

Example Data Link Protocols

• HDLC – High-Level Data Link Control
• The Data Link Layer in the Internet

High-Level Data Link Control

<table>
<thead>
<tr>
<th>Bit</th>
<th>8</th>
<th>8</th>
<th>8</th>
<th>&gt;0</th>
<th>16</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Frame format for bit-oriented protocols.
High-Level Data Link Control (2)

<table>
<thead>
<tr>
<th>Bits</th>
<th>1</th>
<th>3</th>
<th>1</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>0</td>
<td>Seq</td>
<td>P/F</td>
<td>Next</td>
</tr>
<tr>
<td>(b)</td>
<td>1</td>
<td>0</td>
<td>Type</td>
<td>P/F</td>
</tr>
<tr>
<td>(c)</td>
<td>1</td>
<td>1</td>
<td>Type</td>
<td>P/F</td>
</tr>
</tbody>
</table>

Control field of
(a) An information frame.
(b) A supervisory frame.
(c) An unnumbered frame.

The Data Link Layer in the Internet

PPP – Point to Point Protocol

The PPP full frame format for unnumbered mode operation.

PPP – Point to Point Protocol (2)

A simplified phase diagram for bring a line up and down.
PPP – Point to Point Protocol (3)

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configure-request</td>
<td>I → R</td>
<td>List of proposed options and values</td>
</tr>
<tr>
<td>Configure-ack</td>
<td>I → R</td>
<td>All options are accepted</td>
</tr>
<tr>
<td>Configure-nak</td>
<td>I ← R</td>
<td>Some options are not accepted</td>
</tr>
<tr>
<td>Configure-reject</td>
<td>I ← R</td>
<td>Some options are not negotiable</td>
</tr>
<tr>
<td>Terminate-request</td>
<td>I → R</td>
<td>Request to shut the line down</td>
</tr>
<tr>
<td>Terminate-ack</td>
<td>I ← R</td>
<td>OK, line shut down</td>
</tr>
<tr>
<td>Code-reject</td>
<td>I ← R</td>
<td>Unknown request received</td>
</tr>
<tr>
<td>Protocol-reject</td>
<td>I ← R</td>
<td>Unknown protocol requested</td>
</tr>
<tr>
<td>Echo-request</td>
<td>I → R</td>
<td>Please send this frame back</td>
</tr>
<tr>
<td>Echo-reply</td>
<td>I ← R</td>
<td>Here is the frame back</td>
</tr>
<tr>
<td>Discard-request</td>
<td>I → R</td>
<td>Just discard this frame (for testing)</td>
</tr>
</tbody>
</table>

The LCP frame types.