Link Layer protocols
Lecture outline

- The HDLC family
  - High-Level Data Link Control (HDLC)
  - Logical Link Control (LLC)
  - Point-to-point protocol (PPP)
References

- HDLC T 3.6.1
- LLC:
  - Tanenbaum Section 4.3.9
  - Chapter 6 of W. Stallings†: *Local and Metropolitan Area Networks*, 2000
- PPP: T 3.6.2

† Beware of details: Stallings tends to churn out pulp-non-fiction by regurgitating technical minutiae
The HDLC family

“Go back n ARQ ... appears in the various standard DLC protocols, such as HDLC, SDLC, ADCCP, and LAPB. It is not elucidating to know the meaning of these acronyms, but in fact, these standards are almost the same.” [Bertsekas and Gallagher 1992, p. 72]

In the beginning there was IBM ...
1967: IBM’s Binary Synchronous Protocol (Bisync/BSC)
1975: IBM’s Synchronous Data Link Control protocol (SDLC) replaces BSC designed to connect a primary node (mainframe) to multiple secondaries (terminals) using a multidrop/multipoint line

Late 70s:
• 1979: ISO modified SDLC → **High-level Data Link Control (HDLC)** (“Normal Response Mode” of HDLC = SDLC)
• Message Transfer Part Level 2 used for SS7 telephone signalling
• 1976(?): ITU modified HDLC → Link Access Procedure (LAP) for X.25
  1980: LAP-B = LAP updated to reflect modifications to HDLC
  1984: LAPD (D channel of ISDN) [Q.920 and Q.921], LAP-F for Frame Relay, etc

1980s:
• IEEE creates **Logical Link Control** – used with IEEE 802 LANs

Internet:
• Serial Line IP (SLIP – RFC 1055, 1988)
• **Point-to-Point Protocol** (PPP – RFC 1134, 1989)
Outline
High-level† Data Link Control

HDLC is specified in ISO/IEC 13239:2002

A *bit*-oriented protocol:
- Data can be any number of bits in length
  (Most other protocols limit the granularity to bytes)
- Uses bit stuffing to distinguish data from framing sequences (01111110), i.e. source inserts 0 after 5 1s

Frame format:

<table>
<thead>
<tr>
<th>Bits</th>
<th>8</th>
<th>8</th>
<th>8</th>
<th>≥0</th>
<th>16</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>01111110</td>
<td>Address</td>
<td>Control</td>
<td>Data</td>
<td>CRC</td>
<td>01111110</td>
<td></td>
</tr>
</tbody>
</table>

Address identifies secondary node. Primary node is implied.

† Pre-OSI days: Data link was considered to be “high” relative to physical transmission!
T Fig. 3-24
Types of HDLC frames

3 types distinguished by initial bits of Control Field:

0: **Information** frames: carry data with Sequence # and piggy-backed acknowledgement (**Next**)

10: **Supervisory** frames:
   Control data flow and ack when no data on the reverse channel.

11: **Unnumbered**: Connection management and unreliable connectionless transfer.
    Modifier effectively extends type →

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

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

<table>
<thead>
<tr>
<th>Bits</th>
<th>1</th>
<th>1</th>
<th>Type</th>
<th>P/F</th>
<th>Modifier</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(c)</td>
<td>1</td>
<td>1</td>
<td>Type</td>
<td>P/F</td>
<td>Modifier</td>
</tr>
</tbody>
</table>

Shared fields:
- **P/F**: Poll/Final:
  - Primary polls secondary (P/F=P) to invite it to transmit
  - Secondary transmits frames, with last one marked P/F=F
- **Type**:
  - 0: ‘Receive ready’ (RR), i.e. acknowledgement
  - 1: ‘Reject’ (REJ), i.e. negative acknowledgement, go-back-N
  - 2: ‘Receive not ready’ (RNR), flow control through destination pauses source until source receives RR.
  - 3: ‘Selective reject’ (SREJ), i.e. nack, selective repeat

See LLC discussion for an example of these being used.
Unnumbered frames

3b modifier field indicates type:

Connection management:
- **DISC**: Disconnect
- Set mode and reset state:
  - **SNRM**: Set Normal Response Mode (normal in terms of HDLC – abnormal in terms of asymmetrical primary/secondary roles)
  - **SABM**: Set Asynchronous Balanced Mode
  - **SNRME, SABME**: Extended SNRM/SABM with 7b sequence numbers

**FRMR**: Frame rejected, e.g. invalid semantics, despite OK CRC (suggests incompatibility more than does REJ)

Control operation:
- **UI**: Unnumbered information – for “arbitrary information ... for the receiving data link layer itself”? [T p. 237]
- **UA**: Unnumbered acknowledgement – for acknowledging UI and other unnumbered frames
Outline
Logical Link Control (LLC)

Defined in IEEE 802.2
Required by many IEEE 802 MAC protocols, since they don’t provide a multiplexing identifier to identify the higher layer protocols.
In practice, LLC is rarely used with “Ethernet” since most “Ethernet” frames use a pre-IEEE 802.3 format that includes a protocol (SAP) field (instead of a length field)
LLC over Ethernet mainly used for Novell Netware and management/control traffic (e.g. Spanning Tree messages & Nortel’s SONMP)

Differs from HDLC because of multiaccess MAC that provides framing/error detection:
- Has 2 address fields (source & destination) for multiaccess
- Lacks framing delimiters and CRC
  Byte oriented (bit orientation is rarely useful, and MAC layers provide byte-oriented service)
- Sequence numbers grow from 3b to 7b
  Control field for unnumbered frames (lacking sequence numbers) are shorter than for information/supervisory frames

T Fig. 4-24
LLC addressing

802 addressing:
- MAC address identifies a hardware interface ("station")
- LLC Service Access Point identifies a protocol within the device having that hardware interface
- SAPs are 7b: shortage ⇒ SNAP above LLC

Typical network protocol evolution:
- extend one protocol by tacking on parts (in this case another protocol)

Analogous to addressing in the IP/TCP-UDP layers:
- IP address identifies a hardware interface
- TCP/UDP port #s identify a protocol/process within that device.
  (Unlike TCP/UDP, LLC allows multicasting to multiple processes.)
LLC services

Can† provide 3 types of service to higher layers:

- **Connectionless, either**
  - Unacknowledged (“Type 1”), or
  - Acknowledged (“Type 3”). 2 forms:
    - Push: source initiates transfer and receives ack
    - Pull: destination initiates transfer and receives data
      (Service primitives at interface to higher layer:
      - REPLY.request to poll
      - REPLY.indication when response comes
      - REPLY-UPDATE for higher layer to feed data into LLC for responding to polls)
    - Provides acks, but doesn’t provide error recovery, sequencing or flow control. Connection-oriented mode used for this.

- **Connection-oriented (“Type 2”)**
  - Only supports unicast connections
  - Primitives to manage established connections:
    - DISCONNECT a connection (i.e. terminate it)
    - RESET a connection (i.e. reinitialise, possibly losing data – up to higher layers to recover)
    Both include reasons

Priority can be specified for all services.
- This is always passed to the MAC layer to prioritise access (if supported).
- For connection-oriented service, it also prioritises use of LLC resources

† All nodes must support the unacknowledged connectionless service, whereas support of the other services is optional.
LLC data units

**LLC Protocol Data Unit:**

<table>
<thead>
<tr>
<th>DSAP</th>
<th>SSAP</th>
<th>Control</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1 or 2</td>
<td>variable</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Length (B)</td>
</tr>
</tbody>
</table>

DSAP: Destination Service Access Point (6=IP, 170=SNAP, etc)

1st bit indicates individual/group (multicast/unicast)

SSAP: Source Service Access Point

1st bit distinguishes command/response data units

**Control field:**

Bit:  1  2  3  4  5  6  7  8  9  1  1  1  1  1  1  1

- Information: 0  N(S)  P/F  N(R)
- Supervisory: 1  0  SS  0000  P/F  N(R)
- Unnumbered: 1  1  MM  P/F  MM

N(S) = Sequence number(sender)
N(R) = Sequence number(receiver)
P/F = Poll/Final
SS = Supervisory bits
MM = Modifier bits
LLC frame types

Some are restricted to being used only with commands (C), with responses (R) or either (C/R)

For all types of service:
- **TEST (C/R):** loopback test
- **XID (C/R):** eXchange IDentification: types of service supported and receive window size for flow control (C requests, R provides answers)

For **connection-oriented** service:
- As for HDLC: Ready (RR) or not (RNR) or reject (REJ) (all C/R), FRMR (R), SABME (C), DISC (C), UA (R)
- Unlike HDLC:
  - Must operate in SABME mode; no SNRM, SNRME, or SABM
  - Adds “Disconnected Mode” DM (R) to reject a connection request
  - No selective reject

For **unacknowledged connectionless** service:
- **UI (C):** Unnumbered Information (no seq #)

For **acknowledged connectionless** service:
- **P/F bit** distinguishes between pushing (DATA-ACK service) and pulling (REPLY service)
- **AC (C/R):** Acknowledged Connectionless: Permits unnumbered stop-and-wait transmission (in addition to HDLC’s sliding window) (C=data, R=ack)

**Too many details!**
LLC examples

Fig. 6.3 from W. Stallings: *Local and Metropolitan Area Networks*, 2000
Outline
Point to Point Data Link Control

- one sender, one receiver, one link: easier than broadcast link:
  - no Media Access Control
  - no need for explicit MAC addressing
  - e.g., dialup link, ISDN line

- hence the point-to-point protocol (PPP)
Point-to-Point Protocol (PPP)

Currently defined by RFC 1661

Designed to be used when the carrier provides bit†, not frame, transfer, e.g.:
- Routers connected by leased lines
- Home user to ISP over phone line
  (Protocols designed for shared media, e.g. Ethernet, can be used for point-to-point also)

Includes specs for how to test the physical channel ⇒ PPP over Sonet [RFC 2615], etc

So popular, than often layered on top of protocols that provide frame transfer so that these protocols can be accessed by higher layers built to use PPP!:
- PPP over AAL5 [RFC 2364]
- PPP over Ethernet [RFC 2516]

†PPP only sends full bytes, so the lowest-level process is to translate carrier bits to bytes
PPP Design Requirements [RFC 1557]

- **packet framing**: encapsulation of network-layer datagram in data link frame
  - carry network layer data of any network layer protocol (not just IP) *at same time*
  - ability to demultiplex upwards
- **bit transparency**: must carry any bit pattern in the data field
- **error detection** *(no correction)*
- **connection liveness**: detect, signal link failure to network layer
- **network layer address negotiation**: endpoint can learn/configure each other’s network address
PPP non-requirements

- no prevention of long runs of certain bit strings
- no error correction/recovery
- no flow control
- no need to support multipoint links (e.g., polling)

Error recovery, flow control, data re-ordering all relegated to higher layers!
PPP frame format

Flag: code used for byte stuffing

Address & Control: Legacy of the HDLC frame format. An option allows their omission.

Protocol: Which higher layer protocol should this frame be delivered to?
  - e.g. IP or Appletalk; LCP & NCP (PPP control protocols)
  - Link Control Protocol (LCP): Negotiate options & test link

Payload: The good stuff.

“Checksum”: CRC

Flag

† Distinct from the ARPAnet’s original Network Control Protocol
PPP options

Negotiated using LCP:

- Header compression:
  - Omit fields that aren’t used (to save bandwidth over slow lines), e.g. omit Address & Control fields.
  - Length of Protocol field (1 or 2B)
- Maximum payload length (default: 1500)
- Type of CRC (2B or 4B)
- Enable authentication stage?
- Line quality monitoring during normal operation?
# LCP frame types

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

$I = \text{Initiator}$

$R = \text{Responder}$

**Option negotiation during establish phase**

**Terminate phase**

**Responder doesn’t understand**

**Testing, e.g. of line quality**

T Fig. 3-29
PPP major states

Using LCP:
- Test line quality
- Negotiate options

Network Control Protocol (NCP) for IP:
- Called IP Control Protocol (IPCP)
- Protocol field = 8021
- Role: configure/learn IP address

Payload can now be transferred

Done

NCP configuration

T Fig. 3-28