IEEE 802 Logical Link Control

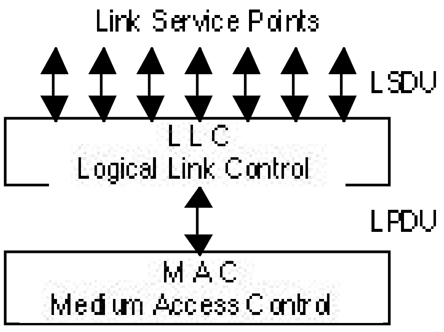
Traditional OSI

The IEEE 802.*x* standards define a series of related Local Area Network implementations. The three layers of the 802 specification are –

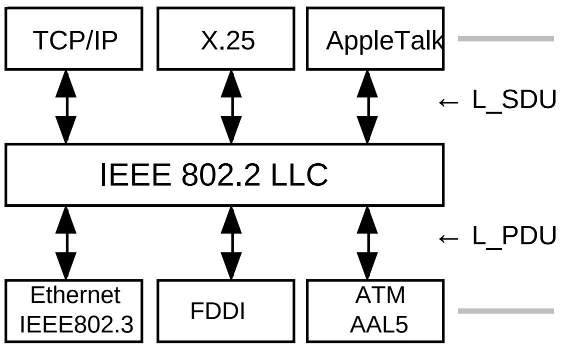
- The Logical Link Control (LLC, 802.2) provides a basic interface for both connectionless and connection-oriented services.
- Medium Access Control (MAC) provides for the transmission of data on the medium and also the defines the protocols which maintain the link operation.
- The Physical Layer provides the actual physical connection for data transfer. The Physical and MAC layers are specific to a network type, but present a common interface to the LLC layer.

IEEE 802.2 – Logical Link Control

This is the upper part of the DataLink layer for the 802.x networks and provides a common interface to the Network Layer from the different media. It accepts *Link Service Data Units* from the Network Layer and delivers *Link Protocol Data Units* to the MAC layer (and the several Link Service Access Points are multiplexed on to the basic service and effectively define sub-addresses for the node.



An actual LLC layer might have several systems above it and several physical protocols



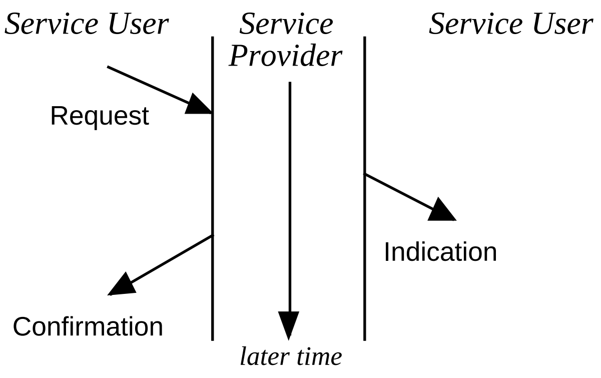
Protocol diagrams

These diagrams indicate the messages exchanged between a service user and a service provider. The basic timing diagram is widely used in data communciations to indicate a time sequence of messages. The basic style of message is well-established in OSI protocols and often carried over into other areas.

The 802.2 LLC protocols use three message types —

- A **Request** is passed down to request a service
- It appears at the "peer" service user as an **Indication**
- A **confirmation** is returned to the requester.

Other protocols may use more message types



Logical Link Control Services

The services are provided by subroutine calls (or equivalent) to the LLC. The most important is the unacknowledged connectionless service, with the two primitives —

$$\left.\begin{array}{r}\text{L\_DATA.request}\\ \text{L\_DATA.indication}\end{array}\right\}$$

with basic functions

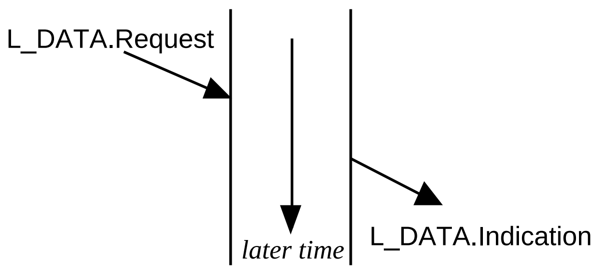
```
        L_DATA.request(local_address,
                       remote_address,
                       l_sdu,
                       service_class)
```

and

```
L_DATA.indication(local_address,
                  remote_address,
                  l_sdu,
                  service_class)
```

Thus the LLC layer is called with a request to pass "`l_sdu`" (user data) from "`local_address`" to "`remote_address`", or to receive `l_sdu` from a remote address. There is no acknowledgement (ie confirmation); the `l_sdu` is just sent and appears.

The protocol diagram is trivial —



The address parameters provide at least the concatenation of the MAC address field and the LLC address field (SSAP or DSAP – Source Service Access Point and Destination Service Access Point). The remote_address for the L_DATA.request may be a broadcast address. The L_DATA.indication returns the identical LSDU as was provided to the matching L_DATA.request.

MAC Service Data Units

The Logical Link Control Layer supplies `m_sdu` (MAC service data units) for transmission by the MAC layer. The format of the `m_sdu` is given later, but uses the primitives –

```
MA_DATA.request(
          destination_address,
          m_sdu,
          requested_service_class)
```

and the corresponding –

```
MA_DATA.indication(
          destination_address,
          source_address,
          m_sdu,
          reception_status,
          requested_service_class)
```

Link Protocol Data Units.

The LLC layer receives information (the LLC SDU) through one of its “Service Access Points” (SAPs), and delivers it to its MAC layer, or vice versa. The information transferred through the network must specify the “Source Service Access Point” (SSAP) and “Destination Service Point” (DSAP); this is held in the LLC header prefixed to the SDU.

For simple traffic the LLC header has the form

DSAP address	SSAP address	Control	Information
8 bits	8 bits	8 bits	8*M bits

The Service Access Points are given “well known” addresses for the usual cases, and the Control byte has the value 0x03 for connectionless data. For data transfer the “Information “ field is the LLC SDU data.

For routed ISO protocols the LLC Header value (in hexadecimal) is 0xFE-FE-03.

For protocols such as IP there is a further level of encapsulation. The LLC Header, with address 0xAA, is followed by a “SubNetwork Attachment Point (SNAP) Header”. The 5-byte SNAP Header has a 3 byte (24 bit) code giving an “administering authority” and 16 bits to identify the protocol (00-80 for IP). The full prefix is then

0x AA-AA-03	LLC Header
0x 00-00-00 08-00	SNAP Header

When transferring AppleTalk the SNAP header value is 0x 08-00-07 80-9B.

In total, an 8-byte header is added to the user message (LLC_SDU) in forming the LLC_PDU submitted to the MAC layer.

Many systems use Ethernet conventions, rather than IEEE 802.3. The 802.3 length *field becomes a* protocol type *field, set to denote the traffic type.*

Other LLC Control values

The control byte (and indeed the whole LLC Header) is based on X.25 conventions. Two other values are used by the connectionless service —

LSB 1	2	3	4	5	6	7	MSB 8	
1	1	1	1	P	1	0	1	XID Exchange IDs
1	1	0	0	P	1	1	1	Test
1	1	0	0	P	0	0	0	UI Unnumbered Info.

The Poll/Final bit (P) may be 0 or 1. (A UI field with P=0 has the value 0x03, as noted before.)

The **XID** command is used to exchange information on the supported LLC types and the receive window size (amount of unacknowledged data).

The **test** command is echoed to show that the link is working.

Note that the bit order is reversed from “normal”, with the least-significant on the left.

Service Access Point Addresses

The address LSB (leftmost) bit is 0 for unique and 1 for group addresses, 00000000 defines a null address and 11111111 is a broadcast address, usually intended for all destination SAPs.

Connection oriented :

This section is not examinable, but is included for completeness and to give you an idea of a standard set of primitive operations.

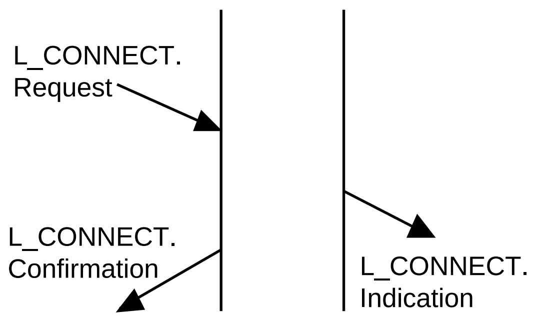
While the 802.2 standard certainly defines a comnection oriented service, this seems to be seldom used. The basic operations are —

$$\left.\begin{array}{r} L\_CONNECT \\ L\_DATA\_CONNECT \\ L\_DISCONNECT \\ L\_RESET \\ L\_CONNECTION\_FLOWCONTROL \end{array}\right\}\left\{\begin{array}{l} .request \\ \\ .indication \\ \\ .confirmation \end{array}\right.$$

The services are very similar to those associated with ISO X.25 communication; L_DATA_CONNECT replaces L_DATA, and a new primitive is added — L_CONNECTION_FLOWCONTOL. (In the connectionless protocols, 802.2 replaces L.UNITDATA by L.DATA.)

Each Link Service Primitive specifies addresses which are as `Dest_Adr.LSAP`. Connectionless data and L_CONNECT may specify a "service class" or priority for the transfer. FLOWCONTROL specifies "the amount of data which may be passed" and may be any agreed value, including zero or infinity.

For all of these services the protocol diagram is like —



The **connection** is initially requested by –

```
        L_CONNECT.request(
                      local_address,
                      remote_address,
                      service_class)
```

which appears at the destination as –

```
        L_CONNECT.indication(
                      local_address,
                      remote_address,
                      status,
                      service_class)
```

and at the source station as –

```
        L_CONNECT.confirm(
                      local_address,
                      remote_address,
                      status,
                      service_class)
```

A **data transfer** is initiated by the command –

```
        L_DATA_CONNECT.request(
                      local_address,
                      remote_address,
                      l_sdu)
```

with the corresponding indication –

```
        L_DATA_CONNECT.indication(
                      local_address,
                      remote_address,
                      l_sdu,
                      service_class)
```

and confirmation –

```
        L_DATA_CONNECT.confirm(
                      local_address,
                      remote_address,
                      l_sdu,
                      service_class)
```

Service disconnection is provided by –

```
        L_DISCONNECT.request(
                      local_address,
                      remote_address)
```

with the remote indication –

```
        L_DISCONNECT.indication(
                      local_address,
                      remote_address,
                      reason)
```

and local confirmation –

```
        L_DISCONNECT.confirm(
                      local_address,
                      remote_address,
                      status)
```

The connection may be **reset**, discarding all unacknowledged PDUs, by –

```
        L_RESET.request(
                      local_address,
                      remote_address)
```

with the remote indication –

```
        L_RESET.indication(
                      local_address,
                      remote_address,
                      reason)
```

and local confirmation –

```
        L_RESET.confirm(
                      local_address,
                      remote_address,
                      status)
```

Finally, the **amount** of traffic may be varied by the **FlowControl** primitive –

```
L_CONNECTION_FLOWCONTROL.request(
                      local_address,
                      remote_address,
                      amount)
```

which has the indication –

```
L_CONNECTION_FLOWCONTROL.indication(
                      local_address,
                      remote_address,
                      amount)
```

The amount may be zero to stop the transfer, or may be set in "implementation specific units".

The LLC Control field may be 8 or 16 bits, defining three basic control classes.

Information Transfer Command/Response (I-Format PDU)	0	N(S)				P / F	N(R)
Supervisory Command/Response (S Format PDUs)	1	0	S S	X X X X		P / F	N(R)
Unnumbered Command/Response (U-format PDU)	1	1	M M	P / F	M M M		

The commands and responses are –

Commands	Responses	
All services		
UI		Un-numbered information
XID	XID	Exchange IDs
TEST		Test
Connection oriented		
I	I	Information
RR	RR	Receive Ready
RNR	RNR	Receive Not Ready
REJ	REJ	Reject
SABME		Set Asynch Bal Mode Extended
DISC		Disconnect
	UA	Unnumbered Acknowledge
	DM	Disconnected Mode
	FRMR	Frame Reject

The "Poll/Final" bit (P/F) may be either 0 or 1 in a command (Poll bit). In the response (Final Bit) it must reflect the value of the Command Poll bit.

The bit codings for these messages are –

Supervisory Command/Response

1 0	0 0	X X X X	P/F	N(R)	RR – Receive Ready
1 0	0 1	X X X X	P/F	N(R)	REJ – Reject
1 0	1 0	X X X X	P/F	N(R)	RNR – Receive not Read

Unnumbered Command/Response

1 1	0 0	P	0 0 0	UI
1 1	1 1	P	1 0 1	XID
1 1	0 0	P	1 1 1	TEST
1 1	1 1	F	0 0 0	XID
1 1	0 0	F	1 1 1	TEST
1 1	1 1	P	1 1 0	SABME
1 1	0 0	P	0 1 0	DISC
1 1	0 0	F	1 1 0	UA
1 1	1 1	F	0 0 0	DM
1 1	1 0	F	0 0 1	FRMR

The entries with a "P" are commands, and those with a "F" are responses.