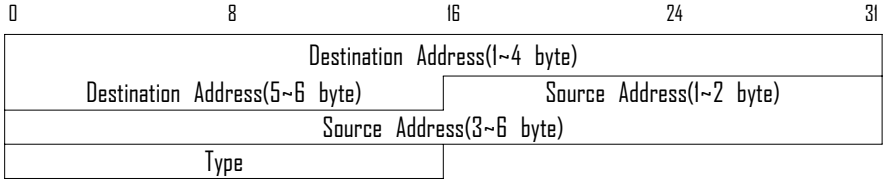
LPX(Lean Packet eXchange) Protocol Specification.

□모든 field 값은 Big-endian(Network byte order)으로 표현한다.

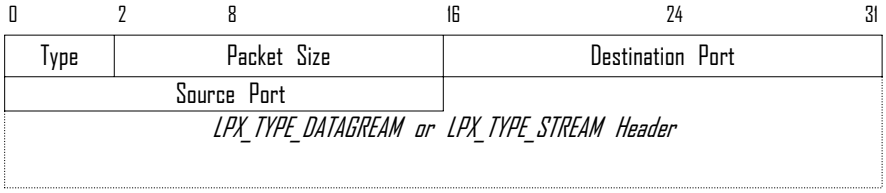
1. Ethernet Header



Type = 0x88ad

2. LPX Header structure.

2.1. LPX Header.



Type = LPX_TYPE_RAW 0x0 // 사용하지 않음
LPX_TYPE_DATAGRAM 0x2
LPX_TYPE_STREAM 0x3

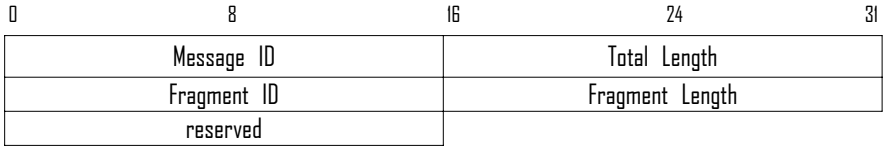
Packet Size = Header size를 포함함 packet의 크기(bytes).

Destination Port = 수신자의 Port 번호

Source Port = 전송자의 Port 번호

LPX헤더는 Type의 값에 따라 아래의 헤더를 가진다. 각각의 타입에 따른 헤더는 10Bytes로 크기가 일정하다. (3장 참조)

2.2. DATAGRAM Type Header.



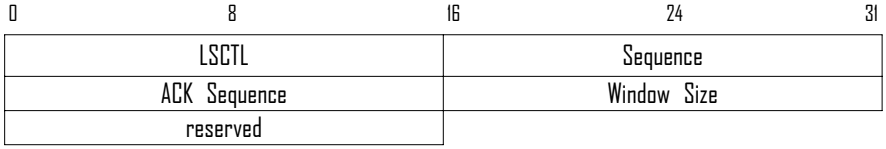
Message ID = Message 고유번호

Total Length = Datagram 전체 크기

Fragment ID = Message가 여러 조각으로 나누어졌을 때 순차적으로 붙는 번호

Fragment Length = 조각난 패킷의 크기

2.3. STREAM Type Header.



0	8	16	24	31
LSCTL		Sequence		
ACK Sequence		Window Size		
reserved				

LSCTL(Lpx Stream ConTroL Bits) =

LSCTL_CONNREQ 0x0001

LSCTL_DATA 0x0002

LSCTL_DISCONNREQ 0x0004

LSCTL_ACKREQ 0x0008

위의 네 가지 flag는 한 패킷에서 2개이상 같이 setting될 수 없다.

ACKREQ를 제외한 세 가지 패킷은 Sequence를 1씩 증가시키고 ACKREQ는 증가시키지 않는다.

LSCTL_ACK 0x1000

암묵적으로 모든 패킷은 ACK를 포함한다.

Sequence = 현재 packet의 sequence.

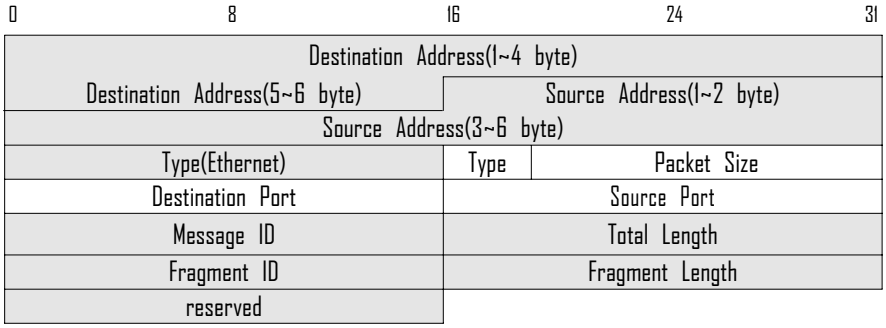
ACK Sequence = ACK의 sequence.

ACK Sequence는 상대방으로부터 받은 다음 패킷의 Sequence이다.

Window Size = 사용하지 않음.

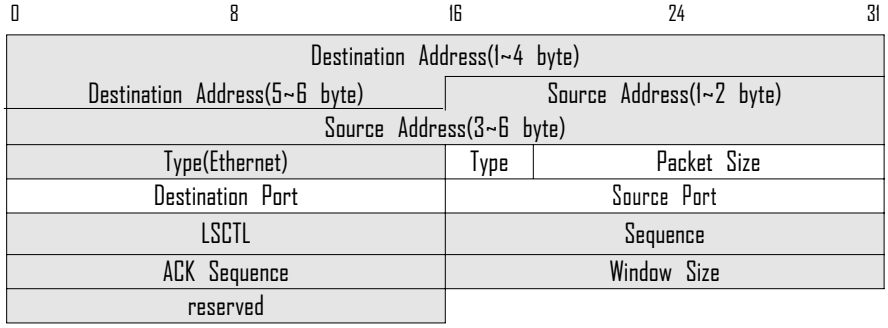
3. Ethernet + LPX(Stream Type)

3.1. Datagram Type



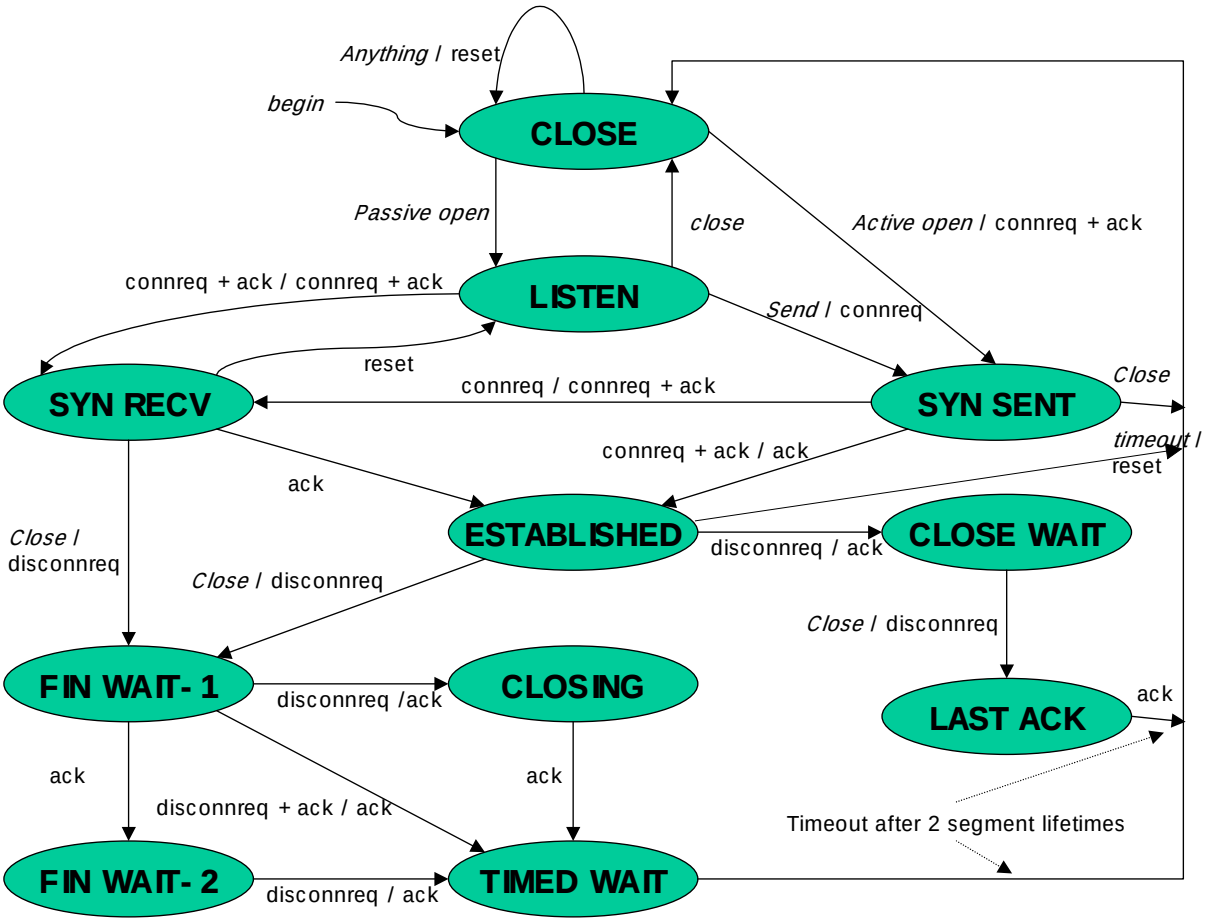
0	8	16	24	31
Destination Address(1~4 byte)				
Destination Address(5~6 byte)		Source Address(1~2 byte)		
Source Address(3~6 byte)				
Type(Ethernet)		Type	Packet Size	
Destination Port		Source Port		
Message ID		Total Length		
Fragment ID		Fragment Length		
reserved				

3.2. Stream Type



0	8	16	24	31
Destination Address(1~4 byte)				
Destination Address(5~6 byte)		Source Address(1~2 byte)		
Source Address(3~6 byte)				
Type(Ethernet)		Type	Packet Size	
Destination Port		Source Port		
LSCTL		Sequence		
ACK Sequence		Window Size		
reserved				

4. LPX Stream finite state machine

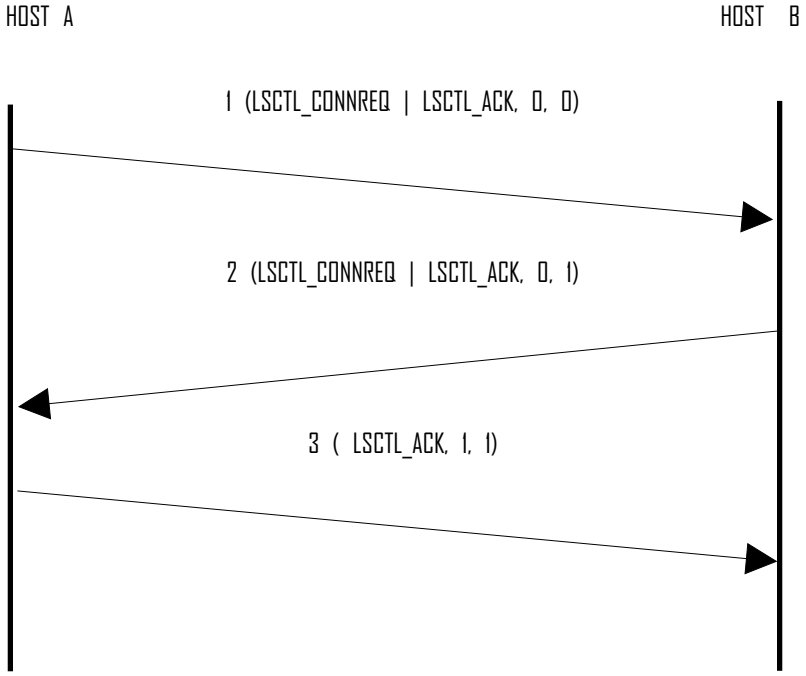


LanDisk는 수동적으로 동작한다. 즉 connecting과 disconnecting은 Host에 의해서 이루어진다. LanDisk는 connection 요구를 받기 위해서 passive open을 수행한다.

5. Connecting

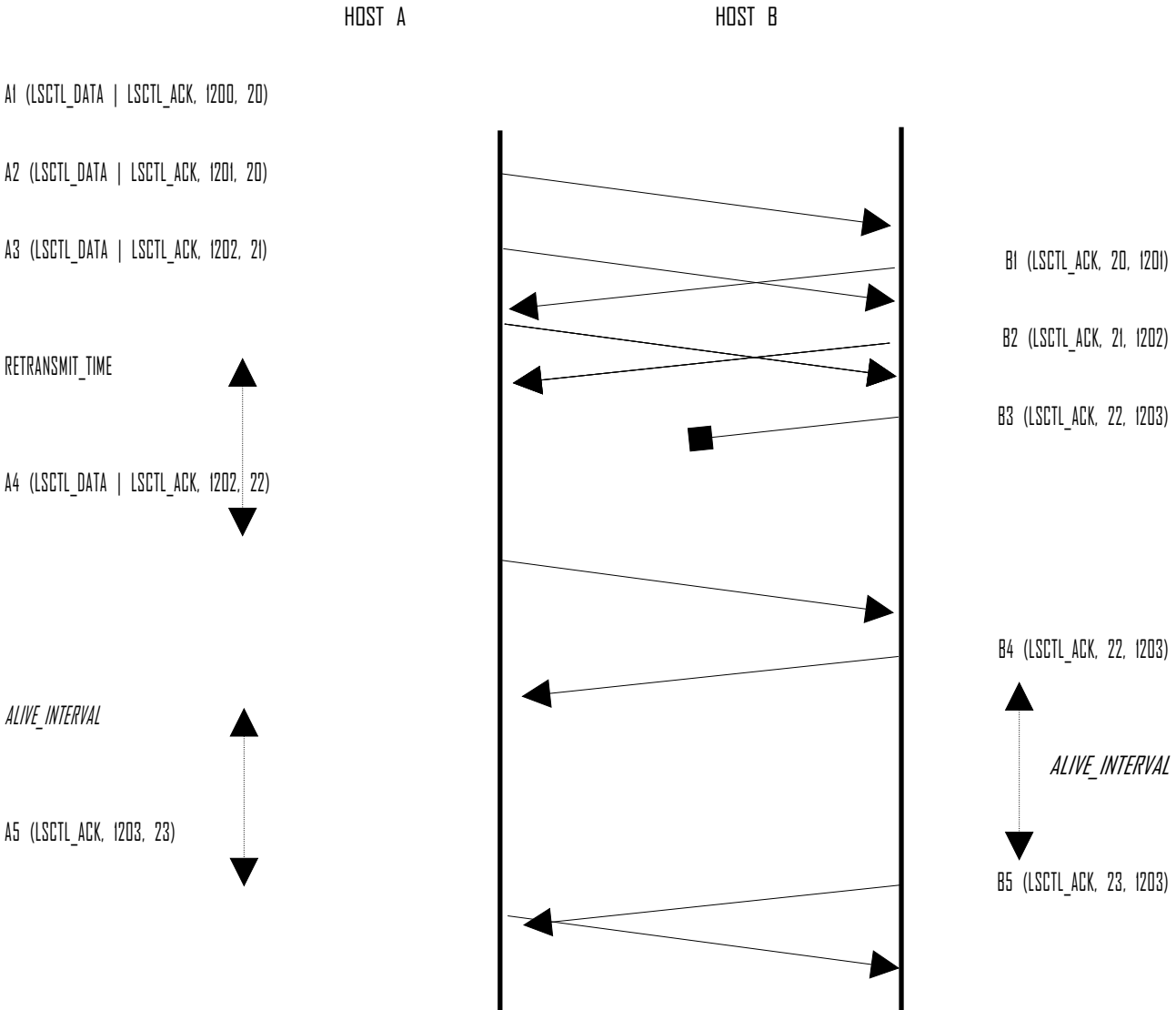
LPX Stream에서 connection을 만드는 방식은 TCP와 유사한 3-way handshaking으로 구현되었다. 아래의 예제는 HOST A가 HOST B에게 connection을 시도하는 것이다. 각각의 패킷은 LPX Stream Type 패킷중의 일부 값을 나타낸 것으로 (LSCTL, Sequence, ACK Sequence)를 나타낸다. LSCTL필드의 값은 flag들을 |(or)연산으로 구성된다. 아래의 예제와 달리 HOST A의 sequence가 100이라면 각각의 패킷은 (LSCTL_CONNREQ | LSCTL_ACK, 100, 0), (LSCTL_CONNREQ | LSCTL_ACK, 0, 101), (LSCTL_ACK, 101, 1) 이 된다.

LanDisk system 환경의 경우 LanDisk는 언제나 수동적으로 동작하므로 예제의 HOST B와 같이 동작한다. 초기화 된 LanDisk는 LISTEN상태이다. host로부터 connection요구(LSCTL_CONNREQ | LSCTL_ACK)를 받으면 SYN_RECV상태가 되고 host에게 ack 패킷을 보냄으로서 ESTABLISHED 상태가 된다.



6. Data Transferring

LPX Stream Type는 상대방으로부터 전달된 패킷의 sequence값이 바로 전(Most recently)에 받은 패킷의 sequence값보다 1이 큰 패킷만을 유효한 패킷으로 인정한다. 그리고 유효하지 않은 패킷은 아무런 조치를 하지 않고 무시한다. 단 ACK 패킷의 경우 sequence값이 1보다 크면 유효한 패킷으로 본다.
LPX Stream Type 상태가 ESTABLISHED일 때 데이터를 전송할 수 있다. 데이터가 담긴 LPX Stream Type 패킷은 LSCTL의 값이 (LSCTL_DATA | LSCTL_ACK)이다. 즉 LPX Stream Type Header의 ACK sequence값이 유효(ACK piggy-bag)하고, LPX Stream Type Header 이후의 값은 데이터를 나타낸다. ACK sequence 값을 통하여 자신이 재전송 해야 하는 패킷이 무엇인지 알 수 있으므로 현재 구현된 LPX Stream Type는 ACK가 포함된 패킷을 받은 경우 아직 acknowledge가 되지 않은 패킷을 재전송 한다.
유효한 DATA 패킷을 받은 HOST는 받은 패킷을 처리하고 상대방 HOST에게 ACK를 전달한다.

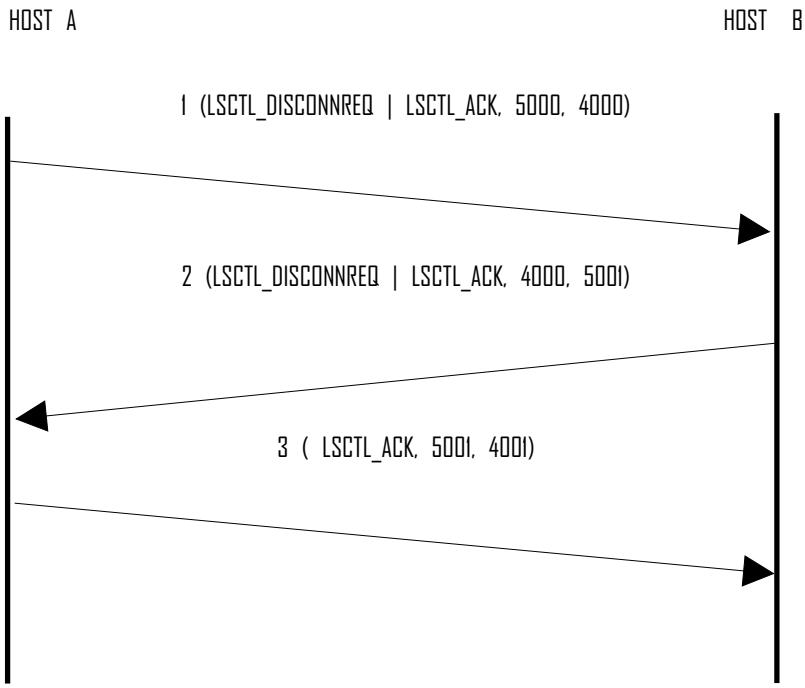


위의 예제는 HOST A가 HOST B에게 세 개의 패킷을 보낸 상황이다. 패킷 A4는 패킷 B3가 사라졌기 때문에 재전송 된 것이다. A5와 B5는 connection이 idle한 상태일 때 상대방에게 자신의 존재를 알리는 패킷이다.
Connection이 비정상적으로 끊어지는 경우는 두 가지이다. 첫 번째는 재전송을 최대 재전송수(10 혹은 20) 만큼 했는데도 그에 대한 ACK가 도착하지 않은 경우이고 두 번째는 idle한 상태에서 상대방으로부터 아무런 패킷도 받지 못했을 때이다. 두 번째 경우의 구현은 connection당 alive_count라는 변수를 두어 상대방으로부터 아무 패킷이라도 받으면 그 값을 0으로 하고, 자신이 idle할 때 ACK 패킷을 보낼 때

마다. 1씩 증가시키는데 그 값이 최대값(10)을 넘으면(Alive_interval * 10의 시간이 지나도 상대방으로부터 패킷이 도착하지 않으면) connection이 끊어졌다고 간주하는 것이다.

7. Disconnecting

LPX Stream의 Disconnecting은 connecting때와 비슷하게 3-way handshaking으로 동작한다. 아래의 예는 HOST A가 HOST B에게 Disconnecting을 요구하는 것이다. 이럴 경우 HOST B는 패킷 1을 받으면 CLOSE_WAIT상태로 되고 상위 layer에서 close를 call할 때 패킷 2번을 보내어 LAST_ACK 상태로 간다. 그리고 패킷 3번을 받으면 connection을 제거한다.



Host A와 HOST B가 동시에 disconnection 요구를 할 수도 있다. 이럴 경우에는 각각의 호스트는 FIN_WAIT-1, TIMED_WAIT상태로 변화하며 위의 그림과 같이 패킷을 주고받는다.