데이터 통신 강의 정리노트

- 본 수업 교재: fifth-edition-data-communications-and-networking

2020124036

교수님이 말씀하신 데통의 특징

- 뒷내용을 알아야 앞내용을 이해할 수 있다.
- 수식을 쓰지 않는다.
- 정성적인 접근을 배우고, 구현을 가르치진 않는다.

통신이론, 디지털 통신과 다르다.

※강의 내용을 임의로 순서를 바꿔서 정리했다.

CH.1 INTRO

- 보이스 vs 데이터통신

1. 옛날에는 보이스(signal)을 단순히 전달했다.

    > - 유선전화를 생각하면 된다.
    > - 따라서, 통화중엔 회선을 독점하고, 독점하다 보니 실시간성이 보장된다. (신호가 계속해서 실시간으로 선로를 타고 가기 때문.)

2. 비교적 최근에는 데이터 자체를 보낼 필요가 있다.

    > - 보이스도 포함하고, 문자, 소리, 영상 등등 다양한 정보(데이터)를 보내게 됐다.
    > - 실시간성은 필요 없게 됐다. (꼭 그런 건 또 아니지만....)
    > - 회선을 여러 사용자와 공유할 수 있게 됐다.

데이터 통신을 하다 보니 상호간 통신에서 이용할 규약이 필요했다.

- 모르스 부호를 생각해보자. 각 신호들이 어떤 뜻을 가지는 지, 미리 약속해놨다. 비트열인 데이터도 마찬가지로 정해놓았다.
- 그뿐만 아니라, 점차 에러의 발견, 수정 등 ...(추가예정)

CH.2 Protocol Layering

- 통신 프로토콜이 어떤 계층으로 구성되어있는지,

- 왜 필요한지,

Protocol

> 통신을 하기 위해 거쳐가야 하는 단계들을 말한다. 일종의 상호간 정한 규칙이다.

Layering

> 계층.
>
> - 프로토콜이 계층화 되어있다는 것이다. -실제로 여러 규칙들이 각 계층마다 있는 형태이다.

Protocol Layering model은 여러개가 있다. 그 중 대표적으로 TCP/IP Protocol을 가장 많이 쓴다.
아래와 같다.
(데통에서는 주로 L1, L2까지 배운다. 2학기 과목 컴정망에서 상위 Layer를 배운다.)

Layer	name(past)	name(present)
L5	Application	Application
L4	Transport	Transport
L3	Internet	Network(N/W)
L2	N/W interface	Data link
L1	H/W Devices	Physical

찾아보니 physical layer을 포함시키지 않는 것 같기도 하고...
암튼 교재는 physical layer도 포함했다.

1. 기본적으로 사용자들은 응용계층(Application layer)에서 응용 프로그램들을 통해 다른 사용자에게 데이터를 보낸다.
2. 한 단계 아래 layer로 내려보낸다. 각 layer에서는, 데이터를 전송하기 전에 header라는 추가 정보를 데이터 앞에 붙인다.
3. 계속해서 아래 layer로 내려보내면서 각 layer에서 붙여준 header들이 붙게 된다.
4. 최종적으로 physical layer에서는 데이터를 실제 보낼 수 있는 형태인 signal로 바꾼다.
5. 받을 때는 역순이다.

> - 받을 때 역순이란 얘기는, 같은 Layer에 대해서는 같은 기능을 수행하는 대칭구조라는 말과같다.
> 다만, 서로 반대되는 기능(전송시에는 header를 붙이고, 수신시에는 header를 제거하는 등....)을 한다.

> - 유선(전송선로 - 구리선, 광선 등)이나 무선(공기중 전파!)으로 보내준다.

> - 비트열 앞에, 추가 정보를 담은 비트열을 이어 붙인다고 생각하면 된다.
> - Encapsulation이라고 부른다. (수신자가 받는 데이터는, 반대로 layer들을 올라오면서 header를 뗀다. Decapsulation이라 한다.)

> - message나 file로, 실제 컴퓨터에서는 비트열로 이루어져있을 것이다.

- 최종 사용자(APP. Layer에서)는 하위 Layer 구현을 알 필요가 없다.
  암호화/ 복호화 등 데이터가 거치는 처리과정들을 알 필요 없이 단순히 전송/ 수신할 수 있다.
- 각 Layer입장에서도 마찬가지이다.
  - 상위 Layer에서 내려온 데이터 - 보통 '패킷'(일종의 단위)이라고 부른다.- 를 받아 정해진 처리를 한 뒤, 전송할 뿐이다.
  - 논리적 연결(logical connection), 혹은 layer to layer commu.라고 부른다.

    > 물론, 실제로는 계속 Layer를 내려가다, Physical Layer에서 signal의 형태로 전송하지만,
    > 각 Layer입장에서는 각자의 Layer에서 전송하는 것처럼 느낀다.

    > 실제로는 data전송이 일어나지 않지만, 개념적으로 사용자가 느끼기에는 data전송이 일어나는 것이다.
    > Physical Layer에서도 마찬가지로 logical connection이 있지만, 더이상 하부레이어가 없기 때문에 physical conenction이기도 하다.

Protocol Layering이 필요한 이유

보이스보다 데이터를 송신하면서 따라야하는 통신 프로토콜(규약)이 복잡해졌다.
이때 구조화해서 계층을 나누면 체계적으로 sys를 구현/ 관리할 수 있다.

> 인터넷에서 배우기로는 문제가 생기면 해당 Layer로 문제범위를 좁힐 수 있어서라고 배웠다.

Protocol과 N/W

보통 Network(이하 N/W, 망)는 공통된 protocol을 사용하는 end host(device)들 -송신, 혹은 수신하는 장비-, 중간중간 거치는 스위치나 라우터 모드를 묶어 부르는 말이다.

> 통신하고 있는 장비들끼리 한데 묶어 부르는 말이다.

같은 망 안에서는 Data Link Layer에서 전송된다.
반면, 서로 다른 망끼리는 라우터에서 IP address라는, N/W Layer의 프로토콜에서 사용하는 주소를 통해 이어준다. (주소얘기는 뒤에 또 나올 예정.) 그렇게 다른 N/W안으로 들어오면, 다시 라우터에서는 IP addr.를 통해 DLL addr를 찾아 header에 붙여 보내준다.(다른 스위치에게 보내준다.)

Layer마다 데이터 단위가 다르다.

데이터는 각 Layer마다 헤더가 붙고, (혹은 떼지고) 그 명칭도 각각 달라진다.

Layer	unit
App.	messages (뜻 있는 문자, message 말하는 거 같다?)
Transport	segment or user datagram
N/W	datagram
Data Link	frame
Physical	bits

나중에 배우겠지만, 각 데이터 단위 - 를 한번에 묶어서 패킷이라 부른다.- 를 주소를 붙여 수신할 기기로 보내는 형식이다.

주소

중간중간 스위치나 라우터가 주소를 보고 길을 찾아준다. 주소는 Layer마다 다르다.
(Physi. L에서 addr.는 없다.) 스위치에서는 Data Link Layer addr.를, 라우터에서는 N/W addr.를 쓴다.

> - 스위치는 동일 N/W 안에서 길 찾아갈 때,
> - 라우터는 서로 다른 N/W 간에 길 찾아갈 때.

똑같이 스위치나 라우터에서도 En/Decapsulation이 이루어져
header를 떼면서 주소를 판별하고, 다시 header를 붙여 올바른 경로로 보내는 과정이 이루어진다.

※교재 그림에서 link는 전송 선로(Physic. L)말고 Link Layer의 link(스위치들로 연결된 망의 일부분)를 의미하기도 한다.

- 동일 회사망(SKT, KT...), 이더넷, Wireless LAN 등등... 스위치로 이어져있다.

A에서 보낸 패킷이 B에 도착하는 예시를 들자면, A에서 보낸 패킷이 media(air or link)를 타고 전송되면서,
스위치들을 지나고, 라우터에서 라우팅(IP addr.를 보고 올바른 N/W로 보내줌)되고,
B가 속한 N/W에 간 뒤(해당 N/W에 속한 라우터에 도착), 거기서 다시 스위치를 지나고, B의 주소로 도착한다.

- Data Link Layer에서는 D.L addr.를 보고 같으면 통과, 상위 Layer로 올려보낸다.
- N/W Layer에서는 IP addr.를 확인하고 같으면 통과, 상위 Layer로 올려보낸다.
- Transport Layer에서는 전송과정 중에 에러가 발생했는지 확인하고, 에러가 있으면 고치거나 재전송 요청한다.
  통과하면 상위 Layer로 올려보내서 최종 수신자가 받게 된다.

Multiplexing

`여러 소스들을 입력에서 받아 1개의 output으로 출력하는 것(송신측)`

> Demultiplexing은 그 반대.

- 하나의 input을 여러 output으로 출력(수신측)

예를 들어,
우선, TCP/IP Layer에서 동작하는 통신 프로토콜 중 TCP와 UDP가 있다.
APP. L.에서 어떤 성능을 요구했냐에 따라, TCP와 UDP 중에서 1개를 고를 것이다.
D.L.L에서는 TCP/IP Layer에서 내려온 패킷이 TCP인지, UDP인지 구별할 필요가 없다.
다만, 본인 Layer(여기서는 D.L.L)에서의 정해진 내용에 따라 header를 붙여 아래 Layer로 내려보낼뿐이다.

- TCP든, UDP든 (여러 소스들을) 별개의 과정이 아닌, D.L.L의 header를 붙이는 작업만 해주고 내려보내는 것(하나의 output으로 보낸다.)이다.
- 반대로 DeMUX는 Layer을 올라오는 패킷을 UDP에 맞게, TCP에 맞게 개별적 처리를 해주게 된다.

TCP: error free를 요구하는 프로토콜이다.
UDP: error를 감수하고 빠른 전송(real time)을 요구하는 프로토콜이다.

L5	HTTP	FTP	DNS	SNMP
L4	TCP		UDP	
L3		IP		

요런 느낌이다.

Append - OSI 7 Layer model
OSI model도 있다.
TCP/IP보다 체계적인 모델이지만, 실제로 사용은 잘 안 한다.

- 70's TCP/IP, 80's OSI.

Layer	OSI	<=name=>	TCP/IP	Features
L7	Application		Application	
L6	Presentation		Application	
L5	Session		Application	
L4	Transport		Transport	IP, Some helping protocols
L3	N/W		N/W	IP, Some helping protocols
L2	Data Link		Data LInk	Underlying LAN, WAN, Wireless protocols
L1	Physical		Physical	Underlying LAN, WAN, Wireless protocols

---

CH.3 Introduction To Physical Layer

- Physical Layer에서 이루어지는 전송 방법, 매체 등을 배운다.

데이터(=정보)를 실제로(Physical Layer에서) 전송할때는

- 아날로그: continuous
- 디지털: discreate

신호(signal)형태로 보낼 수 있다.

주기신호와 비주기 신호로 나누기도 한다.

- periodic

  > - simple: 하나의 signal
  > - composite: 여러 sine wave가 합쳐져 만들어진 periodic한 아날로그 신호.
  >   - 서로다른 freq.의 신호들로 한꺼번에 여러 정보를 담아 보내기도 한다.
  >   - `여러 중첩 sine wv에서는 freq.를 domain으로한다.` time domain으로 하면 보기 힘들어....

- non periodic

아날로그신호는 물리계층의 시간에 따라 변하는, 실수값을 가지는 물리계층의 전기신호.(빛신호일 수도 있지)

아날로그 신호(sine wave)는 time domain에서 세가지 parameter를 가짐.

- amp., fre., phase.

  > amplitude: 진폭

`이 세가지 parameter 차이로 정보를 표시해서 보낸다.` ※ 참고 periode: 주기 phase: 기존 시작점에서 얼마나 멀어졌는지. (x축이 각일 때, 몇도만큼 x축방향으로 움직였는지)

- 중요한 특성

1. time domain에서 'nonperiodic'인 signal은 Freq.domain에서 continu.amp를 가진다.
2. time domain에서 'nonperiodic'이면서 'constrained', 즉 급격한 signal의 변화(pulse)가 있으면, 무한대의 freq. 성분을 가진다.

   > white noise 성분이 있다고 볼 수 있다. dominant한 amp.성분 signal이 있어 무시할 순 있지만, 어쨌든 noise가 존재하는 것이다.

- Bandwidth

  > - 특정 signal의 주파수 성분들이 존재하는 범위. freq. 범위이다.
  > - $f_{max} - f_{min}$

디지털신호는 신호의 LV(주로 signal의 세기)을 discreate하게 나눠 표현한다.
도중 노이즈로 인해 각 LV차이가 작아지면 error가 발생할 수도 있다.

- Bit Rate 는 '초당 몇 비트를 보내는지'이다.

  > Analog signal의 freq.를 생각하자. 단위는 bps(bits per second)이다.

예제 3.19:

Example 3.19 A digitized voice channel, as we will see in Chapter 4, is made by digitizing a 4-kHz bandwidth
analog voice signal. We need to sample the signal at twice the highest frequency (two samples
per hertz). We assume that each sample requires 8 bits. What is the required bit rate?

Solution
The bit rate can be calculated as
$2 \times 4000 \times 8 = 64,000 bps = 64 kbps$

- Bit Length 는 '한 비트가 전송 선로 상에서 차지하는 거리'

  > Analog signal의 wavelength를 생각해보자.
  >
  > - $propagation speed \times bit\ duration$

- 3.3.3 Digital Signal as a Composite Analog Signal

  > 디지털 시그널을 구성하는 다양한 성분들은(푸리에 analysis로 다양한 Analog sig.들을 볼 수 있다.) 즉, composite이고, 이때의 bandwidth가 무한대이다.
  > 디지털이라는 건, 값이 갑자기 변하는 impulse 변화가 있기 때문에, 이 impulse를 구성하는 signal들은 모든 freq.성분으로 존재.
  >
  > - 반대로 디지털 신호값이 0이면 0 freq.으로 만들어져있다.

디지털 신호가 만일,
periodic이면 => 기본 f와 정수배 nf로 이루어짐.
non periodic이면 => continuous한 spectrum으로 존재.

실제 상황에선 non periodic한 digital signal을 많이 만난다.

- 3.3.4 Transmission of Digital Signals

> non periodic한 digital sig.를 보낼 때, 어떻게 보낼 것인가?에 대한 챕터이다.

이때, 전송 sig.은 channel이라는 전송 매체(media)의 특성에 따라 보낼 수 있는 bandwidth가 정해져있다. 이를 channel이라고 한다.

- Baseband Transmission:

> digital signal자체로 보낸다. low-pass channel로 보내게 되는데, 이때 low-pass channel이란 0부터 시작하는 bandwidth를 가진 채널을 뜻한다.
> -> 디지털 신호는 f=0인 성분이 당연히 필요하다.
> 실제 펄스인 디지털 신호를 원형 그대로 보내기 위해선 low-pass이자 가장 높은 주파수 성분이 무한대인 channel이 필요하지만, 실제론 불가능하다. 다만, 양 끝 주파수 성분일수록 amp.가 미미해 무시가 가능하다.
> dedicated medium으로, 동시에 양방형 통신이 안된다.

> `- 무한대가 불가능하다 보니, Low-Pass Channel with Limited Bandwidth이 된다.`

In a low-pass channel with limited bandwidth, we approximate the digital signal with
an analog signal. The level of approximation depends on the bandwidth available

> `rough approxi.`
> N bps의 digital signal을 아날로그로 보내고 싶을 때, worst case를 생각해야 한다. 즉, 최대한으로 1/0 값이 변하는 경우를 생각한다. 예시의 경우, N/2이 된다. 1초에 N개 비트 중 1010... 혹은 0101... 이고, 1이 sine wv의 양의 부분이라면, 한주기동안 2개의 비트를 표현하기 때문이다.
> 3 비트열의 표현은 000 001 ... 111 이고, 각각 (f=0, phase=180), (f=N/4, phase=180), ..., (f=0, phase=0)로 필요한 freq.는 f=0, f=N/2, f=N/4이다. 이 중에서 first harmonic인 N/2가 가장 큰 경우이므로,
> 따라서 필요한 BW는 N/2이다.

- Broadband Transmission(using modulation):

> modulation을 통해 디지털 sig. -> 아날로그 sig.로 바꿔 보낸다.
> 높은 freq.의 analog sig(반송자)에 곱해서 보내는 방식이라 다양한 freq.의 반송자를 쓸 수 있다.
> 그러다보니 같이 여러개를 동시에 보낼 수 있다.
> 따라서 Broadband는 전송효율이 높다고 할 수 있다.

modulation이란 디지털 sig.를 전송을 위해 analog sig.로 바꾸는 것이다.
modulation을 하면 bandpass channel을 쓸 수있게 된다.
bandpass channel은 freq.이 0부터 시작하지 않는 채널이다. <-> 디지털은 0이 있어야지 생각해보니까.

lowpass보다 훨씬 available하다. (0부터 시작하는 bandpass channel이 lowpass라 봐도된다.)

> carrier라 부르는 single-freq.를 가진 analog sig.에 곱해진 뒤, bandpass channel을 지나 전송된다.

- 이와같이 통신채널을 통해 sig.를 보낼 때, error(=sig. impairment)가 발생할 수 있다.

1. attenuation: 에너지 손실. medium의 저항 등으로 에너지가 감소한다. 보통 증폭기로 증폭시켜 해결한다.
2. distortion: 수신된 sig.의 모양이 달라짐.(왜곡) 예를 들자면, 서로다른 여러개의 sine wv로 구성된(composite) sig.의 경우, freq.따라 전송속도가 다르다. 이 전송속도에 따른 delay로 인해, 즉 phase 차이로 인해 왜곡이 발생한다.
3. noise: sig.에 노이즈가 껴서 시그널 모양이 달라진다. 전선 electron의 random motion, 외부소스로부터 전자기기 소음(induced noise), crosstalk(주변 wire에서 영향을 끼침) 등등이 있다.

> signal-to-noise ratio (SNR)
>
> $$SNR = \frac{average\ signal\ power}{average\ noise\ power}$$
>
> db로도 나타낸다.
>
> $$SNR_{dB} = 10 \log_{10} SNR$$

> 에너지 감소를 나타내는 단위로 db(decibel)이 많이 쓰인다. 두개의 시그널의 상대적인 세기를 나타내기 위해 쓴다. 증폭시 양수, 감쇄시 음수이다.
>
> $$dB = 10 \log_{10} \frac{P2}{P1}$$
>
> Voltage의 경우에는 10이 아닌 20이 곱해진다.

- 3.5 Data Rate Limits

> 어떤 한 채널에서 얼마나 빨리 데이터를 보낼 수 있는가? (in bps) Data rate는 3개의 요소의 영향을 받는다.
>
> - 이용가능한 BW
> - 사용할 sig.의 LV.
> - channel의 quality (noise)

> 이론적으로 최대 bit rate(bps)를 알려준다.
>
> $$BitRate = 2 \times bandwidth \times \log_2 L$$

> L은 data를 표시하기 위한 LV.의 갯수:
> LV을 무작정 늘리면 reciever가 미묘한 구분을 해내야 해서 전체적인 sys.의 신뢰성이 떨어진다.

1. Noiseless channel: the Nyquist bit rate

이때, Noise의 유무에 따라 다른 법칙을 사용해 계산한다. 2. Noisy channel: the Shannon Capacity

> 역시 noisy channel에서 이론적 최대 bps를 알려준다.
>
> $$Capacity = bandwidth \times \log_2(1 + SNR)$$
>
> 다만 뉘앙스가 다른게, LV이 몇개던지 공식에 따른 Capacity를 넘을 수 없음을 알려주고 있다.

두개 다 써서 각각의 정보를 알아내자.

The Shannon capacity gives us `the upper limit`;

the Nyquist formula tells us `how many signal levels we need`.

- 3.6.1 Bandwidth

bandwidth는 N/W의 performance를 측정하는 특성으로 쓰인다. 그러나 문맥상 다양한 뜻을 지니기도 한다.

1. BW in Hertz
2. BW in bps

> 실제 데이터 전송 속도.
> the number of bps that a channel, a link or even a n/w can transmit.

> a range of freq. contained in a composite signal/ or a channel can pass

둘은 비례하는 관계라고 한다. bandwidth freq.폭이 넓으면 bps도 늘어나는 듯하다. ch4, ch5에 언급한다고 한다.

- 3.6.2 Throughput

> N/W를 통해 data를 얼마나 빨리 보낼 수 있나?

> the bandwidth is a potential measurement of a link;

B.W. in bps는 물리계층에서 최대 보낼 수 있는 데이터양
Throughput 은 더 상위 계층(각각) Layer에서 보내는 (실제)데이터양 the throughput is

an actual measurement of how fast we can send data.
일반적으로 BW > TP

예시:
Example 3.44
A network with bandwidth of 10 Mbps can pass only an average of 12,000 frames per minute
with each frame carrying an average of 10,000 bits. What is the throughput of this network?
Solution
We can calculate the throughput as

$$Throughput = (12,000 \times 10,000)/60 = 2Mbps$$

- 3.6.3 Latency(delay): 소스의 first bit가 출발한 뒤, 소스의 모든 메시지가 도착하는 데까지 얼마나 걸리는가?

we can say that latency is made of four components:

$$Latency = propagationtime + transmissiontime + queuingtime + process$$

> - propagation time: 전기 신호가 가는데 속도나 거리가 다양하다.
> 신호가 가는데 소스에서 목적지까지 가는데 걸리는 시간.
>
> $$propagationtime = distance \,/\, propagation\ speed$$
>
> - transmission time: 통신 채널의 가능한 전송 rate에 따라 걸리는 시간이다. 메시지는 첫비트부터 마지막비트까지 시간이 걸린다는 점과, BW in bps가 전송속도(bitrate[bps])에 비례한다는 점을 고려한 시간이다.
>
> $$transmissiontime = message\ size \,/\, bandwidth$$
>
> - queuing time: 스위치에 패킷이 몰리면 차례를 기다리게 된다. 이때 버퍼(queue)에 기다리는 시간을 말한다.
> not a fixed time.
> - processing delay도 있다. 스위치에서 header를 읽고 다음방향을 계산하는데, 혹은 송수신 프로토콜 중에 걸리는 시간을 말한다.

- 3.6.3 BW-Delay Product

> 전송 link에는 BW도 있을 거고, delay도 존재할 것이다. 둘의 곱이 의미하는 것은 무엇일까?
>
> $$BW(in\ bps) \times delay = BDP$$
>
> 먼저, BW는 media(link)가 1초 당 보낼 수 있는 비트 수이다.
> delay는 처음 sender에서 출발한 패킷이 receiver한테 (패킷 중에서도 제일

> 앞 비트가)도착하기까지의 시간이다.
> 즉, 전송 선로(link)상에 최대한 많은 비트가 들어찰 수 있는 양을 뜻한다.

- 3.6.5 Jitter (delay의 difference?)

> 송신과 수신 사이의 delay가 항상 일정하지 않다보니, 오디오나 비디오 같은 time sensitive한 용도의 경우에는 영향이 클 수 있다.
>
> - 특히 특정 방식에서 패킷은 각각 다른 경로로 목적지에 도착한다.

CH.4 Digital Transmission

information(data)를 보내기 위해선, 실체(sig.)를 보내야 한다.
data 는 전송을 위해 digital sig.와 analog sig.로 표현(혹은 변환)해야 할 것이다. 4장에선 digital data=> digital sig.와 analog data(sig.그 자체이다.)=>digital sig.를 배울 것이다.

4.1 Digital-to-Digital Conversion

CH3에서는 data가 아날로그/ 시그널의 형태이고, 이를 나타내는 sig.도 아날로그/ 시그널의 형태가 있다고 배웠다.

이번 챕터에서는 어떻게 digital data를 digital signals로 표현하는 지 배운다.

> 데이터는 컴퓨터 상의 정보 - 비트로 구현돼있는, 우리가 전달하고 싶은 대상이다.
> 반면에 시그널은 실체 - 비트를 표현할 수 있는, 전달 가능한 대상(실물?)이다.

bit를 sig.로 만들어주는 과정에서, 1bit에 대해, sig. 한 요소를 매칭시키게 된다.
이때, sig. 한 요소의 크기를 결정하는 문제가 있다.

- 그래서 비례상수 r을 사용한다. r은 각 sig.요소가 몇개의 data 요소(1 bit)를 실어나르는지(표현하는지) 비율이다.

> $$r = 1bit \ / \ sig.elements$$
>
> 예시:
> r=1: 하나의 sig레벨이 1 bit를 표현하고 있다.
> r=1/2: 2개의 sig레벨이 1bit를 표현하고 있다.

- data rate(=bit rate)와 signal rate. data rate는 bps이고,
  `sig. rate`(=pulse rate)는 1초당 보낼 수 있는 sig. elements의 수이다. 단위는 `baud` 라고 한다.

> $$S = N/r$$

> (S= signal rate, N= data rate, r= 1bit를 몇 시그널element에 보내는지.)의 관계가 생긴다.
>
> - 목표는 data rate를 증가시키고, sig. rate를 낮추는 것(요구되는 BW를 줄인다.)이다.
> - data element는 사람이고, signal element는 차량이다. r>1이면, 한사람을 초과해서 한 차량에 탑승하는 것이다.
>   최종 목표는 traffic jams를 막기위해, 더 적은 차량에 가능한 많은 사람을 태우는 것이다.
>
> 1. worst case: maximum sig. rate
> 2. best case: minimum sig. rate
> 3. average case:
>
> $$S_{avg} = c \times N \times (1/r)\ [baud]$$
>
> $$N:\ data\ rate(bps),\ c:\ case\ factor$$

- BW
  CH3에서 inforamtion(data)을 담은 digital signal은 nonperiodic하다고 배웠다.
  또한, 이 경우에는 BW가 무한대의 범위를 가진다고도 배웠다.
  그러나 현실에서는 제한적인 BW를 가질 수밖에 없다.
  ==> 대다수의 freq. 성분들은 amp.가 매우 작아 0에 근접하기에 무시할 수 있다.
  `dominant한 freq.성분들로 이루어진 'effective BW'`를 찾아볼 수 있겠다.
  (이후로 digital sig.의 BW를 얘기할 경우, effective BW를 생각하자.)

  > - frequency는 change를 뜻한다. 역도 그렇다.
  > - 더 많은 digital sig.변화(높은 Baud rate)는 더 많은 freq.성분을 필요로 한다.
  >   - 한번에 전송할 파형이 복잡해지고, 따라서 더 많은 freq.성분.
  > - 결과적으로, BW는 필요한 freq. range를 나타내기 때문에, Baud rate(signal rate)와 관련있다.
  >
  > $$B_{min} = c \times N \times (1/r)$$
  >
  > $$N_{max} = (1/c) \times B \times r$$

- 여담: data rate를 증가시키면 transmission의 speed도 증가한다. => 이해 가능.
  sig. rate를 감소시키면 BW requirement도 감소한다. =>

  > 1. sig rate감소는 초당 보내는 sig element(LV)이 준다.
  > 2. 초당 보내는 sig element(LV)이 줄면 초당 change(pulse)도 준다.
  >   - worst case에 대해서 얘기하는 것 같다.
  > 3. 초당 change(pulse)가 줄면, 한번 보낼 sig의 모양이 단순해진다.
  > 4. 한번 보낼 sig의 모양이 단순해지면, freq.성분이 덜 필요하다.

> 5. freq. 성분이 덜 필요하면, BW가 좁아진다.

1. Line Coding:
   상위 L에서 내려오는 data를 encoder를 통해 signal로 바꾼다.

> data는 컴퓨터에서 비트열의 형태이다. 이를 시그널로 바꿔주는 것이다.

> 1. Baseline Wandering:
>   digital signal을 받아서 decoding할 때, 받은 sig. pow.의 연속적인 평균을 계산한다. 이를 baseline이라 부르는데, data element의 값(0/1 어느것인지)을 결정할 때, 이 baseline를 기준으로 높고 낮음을 측정한다. 그런데 0이나 1이 오랫동안 들어오면, baseline이 0이나 1을 의미하는 voltage LV 자체가 될 수 있다. 그렇게 되면 data element구분이 어려워진다.

Line Coding시 발생가능한 문제점들 2. DC components:
sig. pow. avg가 non zero인 경우. 오랜 시간 같은 voltage LV이 들어오면, Fourier Analysis를 해봤을 때, very low ferq.를 만든다.
이 freq.들은 0에 가깝게 있고, 이 주파수들을 통과시키지 못하는 media가 있을 수도 있다.
3. Self-synchronization:
receiver과 sender의 bit intervals가 같아야 한다. 서로의 sync.가 맞아야 하는 문제!
...

- 4.1.2 Line Coding Schemes 대략 다섯개 정도의 categories.

> 1. Unipolar Scheme:
>   모든 sig. 레벨이 time축에 대해 0과 below(or above)이다. polar NRZ에 비해 normalized pow.가 두배라 비싸서 잘 안쓴다고 한다.
>   - ex) unipolar NRZ(Non-Return-to-Zero)

2. Polar:
   voltage가 times축에 대해 위아래로 값이 있다.

   - polar NRZ:
     - NRZ_L(NRZ-Level):
       voltage의 LV이 비트값을 나타낸다.
     - NRZ_I(NRZ-Invert):
       다음번 비트가 0이면 LV이 안 바뀌고, 1이면 바뀐다.(inversion한다.)
       - NRZ-L/I는 모두 avg signal rate가 N/2 Baud이다.

         > 각각 최선:N, 최악:0이어서 그 평균이 N/2

       - 오랫동안 같은 LV만 표현하는 경우에 receiver가 구분 못하는 오류가 생긴다고 한다.(?)
         아마 어디서 끊어야 할지 몰라서 그런 것같다고 조심스레 생각해 본다.(?)
       - NRZ-L and NRZ-I both have a DC component problem.

- RZ(return to zero):
  NRZ의 문제는 sender와 receiver사이의 not sync.였다. receiver입장에서는 언제 비트가 시작되고,
  끝나는 지 모르기 때문인데, 이는 1비트를 표현 중(양, 또는 음의 값)에 0으로 돌아오는 것이다.
  대신 하나의 비트를 표현하는데, 2개의 sig. change를 써야 하는 것이다. 그러다 보니 BW가 넓어진다.

  > DC components가 없어 freq.성분이 S_{avg}에 몰려있는 스펙트럼이다.
  > 다만,
  > The same problem we mentioned, a sudden change of polarity resulting in all 0s interpreted as 1s and all 1s interpreted as 0s, still exists here. (해석?)
  > 그리고 LV차이가 미묘해져 구분하기 힘들 수도 있다.
  >
  > - 최대 DataRate=2N이고, 최악은 0이라 avg. sig. rate=N.

- biphase:
  Manchester과 Differential Manchester 둘 다 일단 무조건 다음 비트때 inversion이 일어남.

A. Manchester:

B. Differential Manchester:

> NRZ-I와 RZ를 합쳤다.
> 다음 비트가 1이면 현재 레벨을 유지하다 한 비트의 절반 duration에서 lv change.
> 다음 비트가 0이면 즉시 lv change 후에 한 비트의 절반 duration에서 lv change.

> NRZ-L와 RZ를 합쳤다.
> 0이면 중간에 떨어지는 LV change를, 1이면 중간에 올라가는 LV change를 보인다.
> 1뒤에 1이 오는 경우, 다음 비트가 시작하기 전에 low LV로 떨어진다. 0 다음 0이면 다음 비트 시작 전에 high LV로 올라간다.

> 아무래도 M이나 Diff. M은 1bit duration 중간에 lv change가 있어서 sync.기능이 있다.
> 각 비트마다 high&low LV이 있어 DC component가 없다. 다만, NRZ에 비해 sig. rate가 2배이다.
> avg sig. rate가 2배이다. = min. BW도 NRZ의 2배이다.

3. Bipolar

4. Multilevel

5. Multitransition

- 4.1.3 BLock Coding sync나 error detection을 위한 여분의 비트(redundancy)를 준다.
  mB/nB coding이라고도 부르는데, m-bit를 n-bit로 바꿔주는(여분 비트를 추가) 것이다.

  > 예로, In 4B/5B, the 5-bit output that replaces the 4-bit input has no more than one leading
  > zero (left bit) and no more than two trailing zeros (right bits). So when different groups
  > are combined to make a new sequence, there are never more than three consecutive 0s.
  > (Note that NRZ-I has no problem with sequences of 1s.)
  > 4B/5B Table을 통해 표를 보고 한다. 표를 찾아보면 연속으로 들어가는 0을 줄이도록 설계했다. 그러다보니 5비트(=32개 경우의 수)중에서도 26개만 쓰고 있다.
  > ascii로 문자를 mapping한 것처럼 시작/중단 신호 등도 설정했다.

- 4.1.4 Scrambling
  sync.를 위해 예를 들어, long zero LV pulses이면, 그 중 1개를 다른 LV로 바꾸는 것이다. Blcok Coding은 비트수가 증가한다면, Scrambling은 비트수가 증가하지 않는다.
  다만, 미리 정한 규칙에 따라 비트열을 변화시킬 뿐이다.

  > 개요: biphase같은 경우, LAN같은 근거리 통신에는 유용하지만 BW가 넓어 장거리에서는 x.
  > We are looking for a technique that does not increase the number of bits and does provide synchronization.
  > We are looking for a solution that substitutes long zero-level pulses with a combination of other levels
  > to provide synchronization. One solution is called scrambling

4.2 Analog-To-Digital Conversion

CH3.에서 봤듯이, digital sig.가 analog sig보다 좋다. camera나 microphone에서 만드는 analog sig.를 먼저, digital data로 바꾸는 방법을 배울 것이다.
그 다음, 바뀐 digital data는 4.1절에서 배웠듯이 digital sig.로 바꿀 것이다.

> analog sig. ==> digital data를 배울 것이다. (digitization)

- 4.2.1. Pulse Code Modulation(PCM)

1. (Time)Sampling: analog sig.를 sampling.
   Ts라는 Time sampling period마다 (Time stamp마다) Analog sig.를 읽는다.
   f_s는 Ts의 역으로, sampling freq.라 부른다. 3가지의 샘플링 방식이 있다.

> 1. ideal sampling:
>   Ts마다 pulse(time period가 0에 매우 가까운, 순간을 샘플링했다는 것.)를 샘플링한다.

> PCM encoder를 통과하며 거치는 process이다. 총 세단계가 있다.

2. Natural sampling:
   아주 짧은 시간 동안 샘플링하는 것이다. time domain에 대한 amp.의 연속적인 변화가 담겨 샘플링된다.
3. Flat-top sampling:
   아주 짧은 시간 동안 샘플링하지만, Ts마다 그 순간의 값으로 샘플링한다. 연속적이지 않고 매번 일정한 값으로 샘플링 된다.

sampling rate

> sampling시에는 Ts, 즉 sampling rate / freq.를 고려해야 한다.
>
> - According to the Nyquist theorem, the sampling rate must be at least 2 times the highest frequency contained in the signal.
>   - 단, 애초에 sig. BW가 limited일때만 샘플링이 가능하다.
>   - 또, sampling rate가 가장 높은 highest freq.보다 2배인거지, analog sig.의 BW보다 2배 인건 아니다.

Nyquist theorem

> - f_{max} 는 T_{min}이다.
> - f를 2배로 sampling한다는 건, 주기를 1/2로 샘플링 한다는 것.
> - 주기 Ts만큼 sampling했다면, 해당 sig.의 amp값이 고정된다. 시계를 예로 들면, 분침이 undersampling시 거꾸러 가는 것처럼 보일 수 있고, oversamping시에는 정방향을 알 수 있다. NQ theorem을 따르면 방향은 모르지만 위상차가 180도나는 구간을(12시와 6시, 등등)지난 다는 것을 알 수 있다.(한 바퀴를 돈다는 것을 알 수 있다.)

2. Quantizing: 샘플된 sig.가 quantized.
   sampling을 거치면 pam sig.이라 부르는 sample을 얻는다. sample의 값들은 실수값도 존재할 것이다. modulation에선 정수값만 쓴다.
   따라서 미리 정해진 값으로 quantizing해준다.

> 1. We assume that the original analog signal has instantaneous amplitudes between Vmin and Vmax.
> 2. We divide the range into L(원하는 LV) zones, each of height Δ (delta). D 5
> 3. We assign quantized values of 0 to L − 1 to the midpoint of each zone.
> 4. We approximate the value of the sample amplitude to the quantized values.

> - It can be proven that the contribution of the quantization error to the SNRdB of
> the signal depends on the number of quantization levels L, or
> the bits per sample nb,
> as shown in the following formula: $SNR_{db} = 6.02n_b + 1.76db$

3. Encoding: quantized values를 비트열로 바꿔 출력한다.

> 이전 과정들에서 sample이 quantized되고, 한 샘플마다 몇비트로 표현할 것인지 결정되면,
> 각 sample들을 nbits code word로 바꿔준다.
> 이때의
> $BitRate = sampling\ rate \times number\ of\ bits\ per\ sample = f_s \times n_b$

+ decoder에서는...

- the code word => pulse(that holds the amp.), as a staircase signal
- pass through a low-pass filter to become smooth analog sig.
- 복원 완.

+ PCM Bandwidth
Suppose we are given the bandwidth of a low-pass analog signal. If we then digitize the
signal, what is the new minimum bandwidth of the channel that can pass this digitized
signal? We have said that the minimum bandwidth of a line-encoded signal is $B_{min} = c \times N \times (1/r)$.
We substitute the value of N in this formula:

$$B_{min} = c \times N \times (\frac{1}{r}) = c \times n_b \times f_s \times (\frac{1}{r}) = c \times n_b \times 2 \times B_{analog} \times (\frac{1}{r})$$

When 1/r = 1 (for a NRZ or bipolar signal) and c = (1/2) (the average situation), the minimum bandwidth is

$$B_{min} = n_b \times B_{analog}$$

This means the minimum bandwidth of the digital signal is nb times greater than the bandwidth of the analog signal.
`This is the price we pay for digitization.`

- 4.2.2 Delta Modulation(DM)
  pcm은 꽤 복잡한 과정이었다. 비교적 간단한 방법이 생겼다.
  이전 샘플 대비 얼마나 차이나는 지를 이용하는 방법이다.
  code words가 없고, 한 비트씩 보낼 뿐이다.

> - modulator: 증가시 1, 감소시 0을 기록한다.
>   이전 값과 비교를 하기 위해 stair maker (unit)과 Delay unit을 통과시킨 sig.와 들어오는 analog sig.를 비교시켜 판단한다. (두 시그널은 1 period만큼 차이날 것이다.)
> - Demodulator: digital data를 받아서, staircase maker와 delay unit으로 anlog sig.를 만든다. 하지만 역시, pcm과 마찬가지로 low-pass filter를 통해 부드럽게 곡선을 만들어 줘야한다.

- 4.3 Transmission Modes Data transmission은 link(physical layer의, 그러니까 실제 전선)로 보낼 때 두 가지 방법이 있다.
  parallel과 serial하게 보내는 방법이다.

Data transmission			
Parallel		Serial	
	Asynchronous	Synchronous	Isochronous

> - 4.3.1 Parallel Transmission:
>   동시에 여러 비트를 그룹으로 보내는 방법이다. n bits라면 n개의 wire를 이용한다.
>   - 속도가 빠른 장점이 있다. 대신 비싸다는 단점이 있다. 따라서 주로 짧은 거리간에 이용한다.
> - 4.3.2 Serial Transmission: 직렬로 비트들을 보내는 방식이다.
>   one communication channel만 필요하다.(병렬은 channel이 n인가보다.)
>   통신은 parallel하게 하므로(보통 메시지를 한번에 보내죠?), 중간에 serial로 conversion해주는 converter필요하다.
>
> 1. Asynchronous Transmission: 특정 패턴으로 한번에 보내는/ 해석하는 단위를 구분한다.
>   예를 들어, 1과 0사이 8비트를 하나의 단위로 결정하는 것이다. 그러다보니 전송하는 한 단위 사이사이에 gap이 있다.
>   - 다만, 새로운 8bits(=1bytes)가 들어올 때마다, (re)sync를 해 1비트씩 구분하는 작업을 해준다. 즉, bytes단위로는 async.지만, 한 byte를 읽는 동안에는 bit 구분을 위한 sync가 있는 것이다.
> 2. Synchronous Transmission: bit stream을 모아 하나의 'frame'이라는 단위로 만들어 보낸다.(여러 bytes)
>   다만, 각 frame들은 gap없이 전송된다.(구분은 receiver에게 맡긴다.)
>   속도가 장점이다. 추가 비트를 넣거나 뺴는 작업도 필요 없다. 따라서, sync.가 async.보다 빠르다.
>   - sender가 frame을 보낸 시간과 receiver가 받는(frame이 도착한 시간)을 알고있어야 한다.
>   - 즉, time sync.를 위한 추가적인 sig.와 msg가 필요해서 HW복잡도가 증가하는 단점이 있다.

> 3. Isochronous: sync.의 경우, data를 받는 데까지 걸리는 시간(delay)가 항상 일정하지 않아 실패할 수가 있다.
> Jitter가 작아야 한다. (특히 오디오나 비디오같은 real-time기반의 전송일 때) 항상 고정된 data rate를 보장하는 방식이다.

CH5 Analog Transmission

CH4에서는 data를 전송을 위해 digital sig.로 바꿔주는 방법을 배웠다.
CH5에서는 data를 전송을 위해 analog sig.로 바꿔주는 방법을 배울 것이다.

CH3에서 배웠던 내용: low pass channel 사용가능=> digital transmission이 바람직하다.
bandpass channel만이 사용가능 => 어쩔 수 없이 analog transmission을 써야한다.

digital data를 bandpass analog sig.로 바꿔주는 건 digital to analog conversion이라 부른다.
low-pass analog sig(data라고도 볼 수 있겠다.)를 bandpass analog sig.로 바꿔주는 건 analog to analog conv.라 부른다.
- 당연히 smaller BW인 analog sig.로 바꿀 것이다.

5.1 Digital-To-Analog Conversion

sine wv는 amp., freq., phase의 3가지 파라미터가 있고,
이 3가지 파라미터 값을 바꿔서 서로 다른 sig.를 만들 수 있다.
=> 3가지 파라미터 값을 바꾼 다양한 sine wv로 digital data를 표현할 수 있다.

Digital-to-analog conversion		
Amp. shift keying(ASK)	Freq. shift keying(FSK)	Phase shift keying(PSK)
	Quadrature amp. modulation(QAM)	

- 5.1.1. Aspects of Digital-to-Analog Conversion 앞서 배웠던 용어들 짚고 넘어가기.

> - data element = 1 bit.
> - sig. element = 특정 상수값. 하지만 자연상의 sig. element는 analog transmission에서 조금 다르다고 한다. analog sig.다보니 LV값이 아니라 그런 것 같다.
> - $S = N \times \frac{1}{r}\ [baud]$
> N은 bps, r은 여기선(analog transmission) $\log_2 L$이다. L은 서로 다른 sig. element의 수이다.
> - BW는 sig. rate도 있고, difference의 뜻도 있다.
> - Carrier signal:
> analog transmission에서는, info. sig.를 실어나르는(carry, base가 되는) high freq. sig.를 sending한다. 이 base sig.를 carrier sig.라 부

> 른다. 혹은 carrier freq.라 부른다. 이 carrier sig.를 digital info.가 바꾸는 (앞서 말한 sine wv의 세가지 파라미터 중에서) 것이다.
> (modulation, or shift keying)

- 5.1.2 Amplitude Shift Keying
  Carrier sig.의 Amp.만 변화시킨다. 보통 2 LV을 쓰는 Binary ASK(BASK)를 쓴다.

  > 1. 원래 digital sig.의 bandwidth가 baseband(lowpass)로 주어지면, digital sig.에 carrier sig.를 곱한다.
  > 2. 그러면 f_c를 중심으로 하는 bandpass영역으로 이동한다. (왠지는 안알려줬다.)
  >
  > $$A\sin(2\pi f_c t) \times B\sin(2\pi f t) = AB\sin(2\pi f_c t)\sin(2\pi f t)$$
  > $$= \frac{AB}{2}\{\cos(2\pi(f_c + f)t) - \cos(2\pi(f_c - f)t)\}$$
  > $$(f_c\text{을기준으로} \pm f)$$
  >
  > 아마 이런 이유에서..?
  > - BW for ASK:
  >   carrier sig.는 1개인데, modulation 프로세스를 거치면 nonperiodic composite sig. 이 나온다. (=아마 digital sig.)
  >   프로세싱 후 나온 sig.의 BW를 구해보자.
  > - 이런 sig.는 freq. domain에서 continuous한 값들로 나올 것이다. (BW안에서 연속적인 값들을 가질것이다.)
  >   - B_{min}은 S_{avg}임을 앞에서 배웠었다.
  >   - 이때, modulation과 filtering을 거치며 생기는 다른 상수값이 추가된다.
  >   d라는 (0~1사이 값을 가진다고 한다.) 값이다.
  >
  > $$B = (1+d) \times S$$
  > $$S = sig.rate,\ B = BW$$
  >
  >   - BW의 중앙값은 f_c일 것이다.
  >
  >   > 만약 bandpass channel을 사용해야 할 때, f_c를 적절히 고르면 된다는 뜻이다.

- 5.1.3 Frequency Shift Keying
  amp.와 phase는 상수값 그대로지만, 비트값에 따라 f를 바꾼다. 많은 freq.를 carrier로 쓰게 된다. BFSK등이 있다.

  - Binary FSK(BFSK):
    f_1, f_2 carrier 2개를 골라 각각 1과 0에 매칭한다. 이 경우, f_c 2개를 중심으로 하는 2개의 BW가 생기고, 따라서 총 BW는

$$B = (1+d)S + 2Df$$

2Df는 두 f_c의 차이다.

- coherent BFSK: sig. elements 사이에 phase가 continuous하다.
  하나의 voltage-controlled oscillator(VCO)를 사용해, freq.만 바꿔가며 f_c를 선택한 것과 같다.
- non-coherent BFSK: sig elements 사이에 phase가 discontinuous하다.
  BFSK는 두개의 f_c를 골라서 ASK를 두번한 것과 같다. 각각 modulation을 별도로 해준 것이다.
- multilevel FSK(MFSK): f_c를 L개 고르다 보니,

$$B = (1+d) \times S + (L-1)2\Delta_f$$

- 5.1.4 Phase Shift Keying
  amp.와 freq.는 상수로 남고, f_c의 phase가 서로 다른 sig. element를 구분한다.
  오늘날 ASK나 FSK보다 PSK를 더 많이 쓴다고 한다.

  - Binary PSK(BPSK):
    2개의 sig. element를 각각 phase 0°, 180°를 통해 표현한다.
    ASK만큼이나 간단한데다가, 비교적 noise에 크게 영향을 받지 않는다.
  - BW:
    ASK랑 같고 BFSK보다 작다.(BFSK는 f_c 2개로 분리하느라 낭비되는 공간이 있었다.)
  - Quadrature PSK(QPSK):
    1번에 2bits씩 전송하는 방법이다. 총 4가지 sig. element를 표현할 수 있다.
    2bit씩 처리하는데, 그때마다 1bit & 1bit로 나눈 다음, 2개의 분리된 BPSK를 쓴다.
    각 modulator를 통과한 비트들로 sig.가 만들어지고, 총 두개의 성분이 결과적으로 나온다.
    한 성분은 in-phase, 다른 한 성분을 quadrature(out of phase)라고 부른다.

    > 1. 처음 2bits는 serial-to-parallel conversion을 통과해 하나씩 다른 modulator로 들어간다.
    > 2. 각 modulated된 sig.들이 합쳐진다.
    >
    > $$\begin{aligned} &A\sin(wt) + B\sin(wt+90) \\ &= A\sin(wt) + B\cos(wt) \\ &= A'\sin(wt+90^\circ) \end{aligned}$$
    >
    > + modulator로 들어가는 각각의 한 비트의 주기는 2배로 늘어났다.
    > 왜냐하면 각 비트로 분리되기 전에는 10101010...이었다고 치자.
    > 비트가 분리되면서 1_1_1_1...과 _0_0_0_0...으로 분리되기 때문이다.

    - Constellation diagram:
      modulated된 sig. element를 알기 쉽게 보여주는 diagram이다. Y축을 Quadrature 성분의 amp.크기로, X축을 In-phase 성분의 amp.크기로 한다. angle은 phase를 나타내며, Length는 amp.를 나타낸다. => 복소수 평면 느낌 그대로...
- 5.1.5 Quadrature Amplitude Modulation
  PSK는 장비의 phase차이를 구별하는 능력이 필요했다. 이때문에 bit rate가 제한됐다. ASK와 PSK의 장점을 잘 이용해 보자. 가능한 경우의 수가 더 많아진다.

constellation diagram을 이용하자면, 4-QAM은 1사분면에만 나타낼 수 있고,(unipolar NRZ) 모든 사분면에 다 나타나있을 수도 있다.(polar NRZ) 혹은, x/y축 위에 있지 않게, 즉, 두개의 positive LV을 사용해서 (0을 사용안하는 unipolar NRZ) 나타낼 수도 있다.

```
- BW for QAM:
minimum BW은 ASK와 PSK transmission과 같다. 둘의 advantage를 가져왔기 때문이라 볼 수 있다.
```

5.2 Analog-To-Analog Conversion
왜 필요?
각 station에서 만들어내는 analog sig.는 low-pass고, 정부에서 허가해준 BW은 좁고 bandpass인 상황이 있을 수 있다. shifted될 필요가 있다.

- 5.2.1 Amplitude Modulation(AM)

carrier sig.의 amp.가 바뀌도록 modulated. 역시 freq.나 phase는 그대로다.
info.의 variation의 영향대로 amp.만 바뀔 뿐이다.
- AM BW:
f_c를 가운데로 해서, 기존 sig.(modulating sig. 혹은 audio sig.)의 BW의 2배로 modulate된다.

- 5.2.2 Freq. Modulation(FM)

amp.나 phase는 그대로이나 carrier sig.의 freq.가 modulating sig.의 amp.변화에 따라 가도록.
modulating sig.의 amp변화에 맞게 VCO(voltage controlled oscillator)가 출력할 carrier sig.를 바꾼다.
- FM BW:
정확히 계산하기 복잡하나, 경험적으로 알 수 있다.

$$B_{FM} = 2(1 \times \beta)B$$

B는 기존 sig.(혹은 audio sig.)

- 5.2.3 Phase Modulation(PM)

modulating sig.의 voltage lv 변화에 맞춰 carrier sig.의 phase를 바꾼다. carrier의 amp.나 freq.는 그대로 둔다.
FM과 방법이 비슷한데, 한 가지만 다르다. carrier의 주파수 변화가 modulating sig.의 amp.의 도함수에 비례한다.(시간에 대한 변화율) - PM BW:
역시나 경험적으로 알려져있다고 한다.

$$B_{PM} = 2(1+\beta)B$$

CH6 Bandwidth Utilization

link의 BW는 현실에서 제한된 경우가 대부분이다.
어떻게든 효율적으로 BW를 사용해보고자 한다.

6.1 Multiplexing

여러 sig.가 하나의 link를 이용할 수 있도록 하는 방법이다. 하나의 link가 여러개의 channel을 가질 수 있어서 생기는 여러 방법들이 있다.
일종의 MUX와 DEMUX를 사용한다.

- 6.1.1 Freq.-Division Multiplexing(FDM)
  sig.들을 각각 다른 carrier freq.를 가지게 한다. 그러다보니 link의 BW가 보낼 sig.들의 결합(`composite`) BW보다 커야 가능하다.
  - 전송 link의 가능한 BW를 여러 channel로 나눠 각각의 f_c를 각 신호들에 할당해 준다.

sig.끼리의 overlapping을 방지하기 위해 guard bands라 부르는 BW가 각 CH들 사이에 있다.

- +analog sig.를 보내는 방법인데, digital sig.도 analog sig.로 바꾸는 과정이 앞서 있으면 digital sig.도 이 방법으로 전송 가능하다.

- 6.1.2 Wavelength-Division Multiplexing(WDM)
  광선에서 high-data rate를 전송하기 위해 고안된 방법이다.
  data rate는 금속선보다 빠르지만, 광선 하나에 여러 신호를 전송해야 하는 필요가 있었다.
  아이디어는 FDM과 같다. freq.가 서로 다른 sig.들을 MUX로 합쳐서 link로 전송한다.
  다만 이때의 freq.가 매우매우 높다는 점이 다르다.
  실제 방법은 매우 복잡한데, 기본 아이디어는 간단하다.

  > 프리즘을 통해 다양한 freq.인 빛을 휘게 만들어 하나로 합친다.

- 6.1.3 Time-Division Multiplexing
  TDM은 여러 source들이 link의 high BW를 공유할 수 있도록하는 `digital process` 이다.

- digital data가 필요하다. analog data도 digital data로 바뀌는 작업(sampling~)을 거치면 쓸 수 있다.
- 여러 low-rate channel들을 하나의 high-rate channel로 합쳐준다. link의 BW의 일부분을 공유(FDM)이 아니라, time을 time slot들로 나눠 순차대로 각 source들에게 할당해준다.
  (이 장에서는 1-1link를 얘기하고 있으므로, 계속해서 순차적인 time slot들이 한 des.에게 전송된다.)

  - > Example 6.8
    > Four channels are multiplexed using TDM. If each channel sends 100 bytes/s and we multiplex
    > 1 byte per channel, show the frame traveling on the link, the size of the frame, the duration of a

    - 이 경우 하나의 frame에 3개의 time slot이 있다.
      `In synchronous TDM, the data rate of the link is n times faster, and the unit duration is n times shorter.`
    - Interleaving:
      TDM은 MUX쪽이나 DEMUX쪽에서 sync.된 스위치가 각 source들을 순차적으로 연결/해제한다.
      이 process를 interleaving이라 한다.

    synchronous TDM:
    A1, A2, A3...
    B1, B2, B3... C1, C2, C3... 이렇게 3개의 소스에서 data들이 들어오고 있는 상황을 생각하자. 각 data들은 T의 duration을 가지고 있다.
    MUX를 통과한 다음에는 각각 1 frame안에 A1/B1/C1, A2/B2/C2, ... 이렇게 담겨있을 것이다. 그리고 각 frame은 T의 duration을 가지고 있다. (source들에서 온 데이터들은 각 T/3 duration을 가질 것이다.)
    frame, the frame rate, and the bit rate for the link.

    - MUX에서 1bytes/channel 이므로 한번에 4bytes를 frame으로 만들 것이고, frame duration은 source들로부터 오는 duration과 같다. bit rate for link는 8bit= 1byte이고, source bps의 4배임을 이용해 구한다.
    - Empty slots문제:
      보낼게 없어도 고정적으로 frame에는 해당 source의 time slot이 할당돼있어서 빈슬롯을 보내게 된다.
    - Data Rate Management:
      TDM의 문제 중 한가지는, 인풋 source들의 data rate 불일치

이다.
세가지 해결방법이 있겠다.

1. multilevel multiplexing
2. multiple-slot allocation
3. pulse stuffing

> multilevel muxing처럼 어떤 입력들의 전송 rate가 맞지 않을 때 쓴다.
> 다만, 한 source를 demuxing해서 해당 source로부터 frame에 차지하는 slot을 늘린다.

> 인풋하나의 data rate가 다른 인풋들의 배수일 때 다른 인풋들을 mux를 통해 배수로 묶어준다.
> 해당 인풋들이 frame에서 차지하는 slot들을 하나로 만든다.

> 가끔은 source들의 data rate가 서로의 정수배가 아닐때도 있다. 그럴때는 dominant하게 high인 data rate에 맞춰 나머지에 dummy bits들을 넣어준다.

- statistical TDM:
  입력라인에 데이터가 존재하는 대로만 전송한다. input마다 할당된 자리가 없다.
  BW efficiency가 좋다.

  > 1. 대신 addressing이 필요하다.
  >   정해진 자리가 없다보니(sync.는 정해진 자리에 따라 output이 어디로갈지 이미 정해져있다.),
  >   addressing을 위한 추가 bit가 붙어 frame이 커진다.
  >   (누가 보내고, 어디로 가는 지.)
  > 2. 특정 input을 언제보낼 지 모르기 때문에, 받은 data의 순서가 정해져있는 경우,
  >   queuing delay가 생길 수 있다. 먼저 도착한 후순서 frame들이 queue(buffer)에서 대기하기 때문이다.

6.2 Spread Spectrum

Multiplexing은 여러 소스에서 오는 sig들을 효과적인 BW에 맞게 합치는 방법이었다.

- link의 BW가 source들에 맞게 분배됐다.

SS에서도 여러 소스에서 오는 sig.들을 BW에 맞게 합친다.
하지만, SS는 wireless 환경(LAN, WAN...)에 맞게 개발됐다. air medium일 때, 외부 도청이나 방해를 받기 쉽다.
이를 막기위해 spread spectrum techniques와 redundancy를 더한다.

> 1. 각 station(방송국?기지국?)에 할당된 BW(link의 channel)은 필요한 BW(송신할 sig.)보다 커야한다. redundancy를 추가하기 위해서이다.
> 2. original BW인 B가 확장된(폭이 늘어난) B_{ss}에서는, source로부터 original BW가 만들어진 후에 spreading process를 거쳐 만들어져야 한다.
>   - spreading process는 spreading code를 이용해 BW를 B에서 B_{ss}로 넓힌다. spreading code는 숫자들의 나열로, 패턴이 있다. spreading BW는 두 가지 방식이 있다. FHSS와 DSSS이다.

- 6.2.1 Frequency Hopping Spread Spectrum(FHSS)
  modulate될 M개의 freq. Carrier를 고른다.
  - pseudorandom code generator가 k-bit pattern을 every hopping period T_h 마다 만든다.
  - freq. table을 매 hopping period마다 사용해서, pattern으로부터 carrier freq.를 찾아낸다.
  - source sig.가 carrier sig.로 modulate된다.
  - BW sharing:
    hopping freq.의 갯수가 M이면, M channel을 B_{ss}를 이용해 하나로 Muxing할 수 있다.
    이를 이용하면 여러 M개의 station에서 서로다른 M개의 f_c를 고르면 B_{ss}인 하나의 링크를 나눠쓸 수 있다.
    FHSS는 FDM이랑 닮았는데, 차이점은 FDM은 계속해서 시간축을 따라 정해진 f_c를 할당된 source가 계속 쓰는 것이고,
    FHSS는 매 hopping period마다 f_c가 바뀔 수 있다.
- 6.2.2 Direct Sequence Spread Spectrum(DSSS) 똑같이 original sig.의 BW를 확장시키는 방법이지만, 각 data의 bit를 spreading code를 이용해서 n bits로 바꾼다. (n개의 bit들은 chip이란 단위로 묶어 부른다.)
  BW가 n배 커질 것이다. spread sig. rate도 original sig.rate의 n배가 된다.
  BW sharing은 될수도 있고, 안될수도 있다.

6.2.3 Orthogonal Freq. Division Multiplexing (OFDM은 교재엔 없지만 교수님이 추가로 넣으신 내용이다. 죄송하지만 설명을 너무 못하신다! ㅎㅎ...)
FDM은 하나의 ori. sig.에 하나의 f_c를 썼다.
MCM(Multi carreir modulation)은 하나의 data에 여러개의 carrier(sub carrier)를 쓴다. - 전송 rate를 높일 수 있다. - 무선 통신에서 많이 쓴다. - OFDM은 MCM의 일종이다.

1. data input을 serial to parallel을 통해 병렬로 만든다.
2. 병렬 data input을 mux로 보내 B_{ss}를 통해 보낸다.

- OFDM:

sub carrier이 많을 수록(freq.효율 좋다.), symbol(하나의 sig. element) period가 늘어나고(전송하는 한 단위가 길어지지.),
반면에 multi path fading은 줄어든다.

> air로 전송하는 도중에 장애물에 의해 반사파가 생기고, 반사파는 비교적 느리게 des.에 도착한다.
> 결과적으로 수신측에서 비트끼리 겹쳐 혼동을 줄 수 있다. (multi path fading)

하나의 정보를 여러개의 densely saparated sub carrier로 나눈다. 이때, 직교성을 이용해 간격을 최소화한다.
그러다 보니 인접 carrier spectrum이 overlapping되지만 문제 없다.

> sub carrier에 대응하는 sig. duration을 T로 통일한다.
> 그러면 서로 null(value: 0)으로 가는 간격이 항상 같아서 overlapping되는 지점값은 항상 0일 뿐이다.

data rate도 줄어들 것이다. 하지만 symbol duration을 줄이지 않는게 OFDM의 중요 목적이다.

- 한꺼번에 촘촘히, 많이 보내다보니 BW를 아끼게되고, error(symbol duraion도 지키는)도 적다. 푸리에변환을 쓰는 이 방식이 고속연산이 가능해 고속전송에도 유리하다고 인터넷에서 그랬다.
  `교수님은 고속전송시 발생할 error를 낮추니 고속전송에 유리하다고 하신다.`

+sub carrier는 결과적으로 하나의 f_c를 쓴 것이다.
0. sub carrier들을 직교성을 이용해 만든다. modulation을 실행한다.

1. freq.-domain QAM을 받아 transmitter에서 inverse furier transform을 수행한다.
2. freq. spectrum에 대응하는 time sample값들을 받아 전송한다.
3. 수신측에선 FFT를 이용해 여러 subcarrier들의 결합신호를 받아 demodulation을 수행한다.

CH.7 Transmission

physical layer을 전 ch에서 배웠다. 이제는 실제 신호가 전송되는 transmission media에 대해 배운다.
미디어는 wire(cable)과 wireless(air)로 나눌 수 있다.

		Transmission media			
guided(wired)			unguided(wireless)		
twisted pair cable	coaxial cable	Fiber-optic cable	Radio wave	Micro wave	Infrared

7.1 Introduction transmission media란 정보를 source에서 des.까지 실어나르는 무엇이든 될 수 있다.

7.2 Guided Media 관형태의 선들을 말한다.

- twisted-pair cable

  > 선 중 하나는 sig.로, 하나는 gnd로 쓴다. 둘의 difference로 sig.의 값을 결정한다.
  > 그러다 보니, 외부의 crosstalk로부터 거리가 같아서, difference는 여전히 유지된다. - Unshielded:
  > 플라스틱 커버 - 외부전파를 막지 못한다.
  > 많이 사용된다. - shielded twisted-pair cable:
  > 플라스틱 커버 안에 메탈커버도 있어 외부영향을 줄인다. EIA에서는 UTP categories를 만들었다.
  > 숫자가 늘수록 data rate가 올라가고, category 3부터는 LAN용으로 쓴다.
  > crosstalk 영향을 줄일수록 (shield가 잘될수록) bit rate가 늘어날 수 있다.
  > 왜냐하면 bit rate가 늘수록 bit duration이 줄기 때문이다.(영향을 잘 받기 때문이다.) - 더 높은 data rate는 - 더 넓은 BW - 따라서 high freq.면 attenuation도 심해지므로,
  > 리피터를 많이 사용하거나, 통신선로를 굵게하거나, shielding을 잘 해야 한다. gauge가 높을수록 직경이 줄고, bit rate에 따른 감쇄(attenuation)이 잘 일어난다.
  > - 생각해보면 직경이 줄수록 선로 자체의 저항이 커질 것이다.

- coaxial cable

  > twp cable보다 higher freq. range.를 가진다. higher일수록 남아있는 안쓰는 BW이 많아서
  > 상대적으로 보다 더 넓은 대역폭을 사용할 수 있게 된다. attenuation도 심해지기 때문에 more repeater 필요하다.

category는 RG 뒤에 붙은 숫자에 따라 구분돼있다.
- RG -59: 75옴 / cable TV - RG -58: 50옴 / thin Ethernet - RG -11: 50옴 / thick Ethernet BNC connecter등을 예시로 들 수 있겠다.

- fiber-optic cable

  > glass / plastic으로된 core를 마찬가지로 재료를 둔 cladding이 감싼 형태이다. 빛 신호를 전달한다.

여러 propagation Modes(전파모드?)가 있다.

	Propagation Modes	
Multimode		Single mode
Step index	Graded index	

- multimode, step index:
  여러 sig.들을 서로 다른 각도로 전송한다. 안쪽 물질 밀도가 일정해서 안쪽에서 반사된 빛의 경로가 직선적이다(?)

- multimode, graded index:
  여러 sig.들을 서로 다른 각도로 전송한다. 안쪽 물질 밀도가 일정하지 않아 안쪽 반사되는 빛의 경로가 휘어지기도 한다.

- single mode:
  모든 sig.들을 거의 일직선(a small range of angles)으로 보낸다.

장점 :

- higher BW: 빛의 특성을 이용했기 때문이다.
- less attenuation: 따라서 멀리 보낼 수 있다. (repeater = regenerator없이도)
- 가볍다 등등...

단점 :

- 유지보수가 힘들다.
- 단방향이다. 양방향을 위해 광선 두개가 필요하다.
- 비싸다.

7.3 Unguided Media: wireless 전파를 이용, air를 통한 전송을 말한다.

3kHz~300GHz의 'Radiowave ~ microwave'와 ~400THz의 Infrared (너머는 Light wv)가 있다.

- 2MHz below: 저주파.
  지구 지표면을 따라서 GND propagation을 한다.
- 2-30MHz: 이온층 반사(Ionosphere에서 반사)
  sky proapgation.
- 30MHz above: line of sight propagation (high f)

VLF	3-30kHZ	gnd	long range radio
LF	30-300kHZ	gnd	navigation
MF(middle)	300k-3MHz	sky	radio
HF	3-30MHz	sky	am
VHF(very)	30-300MHz	sky/lineofsight	fm
UHF	300M-3GHz	lineofsight	cellular phones
SF	3-30GHz	lineofsight	satellite
EHF	30-300GHz	lineofsight	radar

Radio WV v.s. Mircro WV:

radio wv는 omnidirectional antena(전방향 전파),
micro wv는 unidirectional antena(단방향 전파)

- 그러다 보니 micro wv는 좁은 지역에 focusing.

고주파 사용시 전파의 음영지역(닫지 않는 지역)이 생기지만, 오히려 하나의 기지국이 관장하는 Cell(area)가
다른 cell을 침범하지 않을 수도 있고, 남는 BW는 높은 주파수...
고주파에서는 더 높은 data rate를 사용할 수 있다.

- 7.3.3 Infrared(적외선 300GHz above)
  short range로 인해 실내에서 적합하다.

  > 리모컨 등등....

CH.8 Switching

교차로를 생각하면 된다.
교차로는 여러 도로들이 모이는 곳으로, 최종 목적지를 위해 어느 길로 가야할지 생각해야 한다.

- data가 switch에서 뻗어나가는 많은 선로 중 1개를 고르게 된다.

switching은 모든 layer에 대해 적용할 수 있는 개념이나,
이번 챕터에서는 physical. L와 DLL에서 switching에 대해 배운다.

8.1 Introduction
N/W는 통신선로로 연결된 device(node)들의 집합이다.
모든 노드들끼리 1-1연결이 된 full meshed N/W는 비효율적이다.

- 선로를 모두 사용하고 있지 않기 때문. 따라서 선로 하나를 여러 노드가 나눠쓰게 한다.
- 스위치를 통해 최종적으로 보내고/ 받는 end node(device)들을 연결하는 것이다.

이때, 경로는 여러 스위치들을 돌고돌아 갈 수도 있다.

- 스위치에서 어떤 방향으로 가는 선로를 고르는 방법을 스위칭/ 또는 포워딩(forwarding)이라고 부른다.

스위치는,

> 1. 보내는 기능.
> 2. 그걸위한 경로 설정 기능.

스위칭 방식은 circuit, packet, message 스위칭 방식이 있고,
다시 packet스위칭은 virtual circuit방식과 datagram방식으로 나뉜다.

Switching		
circuit switching	packet switching	message switching
virtual circuit approach	datgram	

- 8.1.2 Switching and TCP/IP Layer packet교환이 일어나는 L이 아니기 때문에, sig.들을 phy. L의 스위치들이 한 길따라 갈 수 있도록 하는 기능을 한다.
  - sw. at phy. L:
  - sw. at DLL: packet sw.이 일어난다. 이때, 여기서 패킷은 frame /cell이라 불리는 단위이다. 보통 virtual circuit방식으로 많이 쓰인다.
  - sw. N/W L:
    packet sw.이 일어난다. ~
  - sw. App L:
    message sw.만 있다. 개념적으로, e-mail이 message-switched communication의 일종이라 할 수 있다.

여러 Layer에서 각자의 프로토콜로 스위칭이 발생한다.

8.2 Circuit-Switched N/W physical links들로 연결된 스위치들로 이루어져있는 NW이다. 각각 end node들 사이의(station들 사이) link들은 N개의 채널들로 이루어져있다. (devided by using FDM or TDM)

- station 사이의 link들은 dedicated(전용선로)이다. 나눠쓰는 방식을 이용해 효율을 높인다.

circuit sw. NW는 세가지 phase(단계)가 있다.

- 8.2.1 Three Phases:

1. connection setup - Dedication확보요청 및 수립.

   > set up request와 ack를 스위치들이 나르면서 길을 확정해준다.

2. data transfer
3. conenction teardown

- 8.2.2 Efficiency:
  다른 두가지 방식에 비해 좋지 않다. dedicated라 다른 해당 선로를 이용자들은 deprived(박탈)당한다. 다만, 실시간성이 필요한 통화 연결에 사용했다.

- 8.2.3 delay:
  전화걸때는 setup시에는 오래걸릴수도 있지만, 여기서는 transfer할 때 delay를 말하므로, 실시간성=delay가 적은 장점이 있다.
  중간 스위치에서 길을 확보해놓고 계속해서 연속적으로(패킷, 여기선 신호의 no waiting) 송수신을 하기 때문이다.

8.3 Packet Switching 보내고자 하는 message를 패킷으로 나눠서 보낸다.

- 패킷은 fixed/ variable size로 나뉜다. (프로토콜이 결정한다.)
- 예를 들어, 더 작은 패킷을 쓰는 NW로 패킷이 전송됐으면, 새롭게 쪼개주는 과정이 필요할 것이다.
- 8.3.1 Datagram NW:
  각 패킷들이 독립적으로 다루어지는 NW.
  N/W L에서 주로 이루어진다.
  같은 message에 속해있는 패킷들이 각각 길을 찾아 목적지로 간다.
  - circuit sw에 비해 eficiency가 좋다.
  - 패킷들이 하나의 스위치에 몰리면 스위치의 queue에 대기한다. 각 스위치는 routing table이라는 게 있어서 des. addr.에 대응하는 output port를 가지고 있다.
    즉, des. addr.에 맞춰 해당하는 port로 forwarding (switching)해준다.
  - 따라서 setup이랑 teardown phase가 필요없다.
  - 동적이고(dynamic) 주기적으로 업데이트 된다.
    - circuit sw. table은 setup에서 써지고 teardown에서 삭제된다.
  - 스위치에서는 routing table을 참조하는 시간(processing time)이 필요하다.

각 datagram NW에서는 각 패킷 header에 des. addr.가 있다. 이 des. addr는 도착지에 갈때까지 사라지지 않는다.

> 아마 N/W addr를 얘기하는 것 같다. (iP addr)

virtual-circuit NW에 비해 greater delay가 있다.

- 물론 setup, teardown phase는 없지만, forwarding되는 동안 기다리는 시간이 존재한다.(waiting time)
- > addressing:
  > global과 local addr.가 있다.
  >
  > - global addr. (ex: IP):

  - data link L에서 작동된다.
    - circuit sw.은 physic. L, datagram은 N/W L이었다.
  - circuit sw. nw처럼 똑같이 setup과 teardown phase가 존재한다.
  - 데이터들은 패킷으로 나뉜다. header에 addr도 있다. 다만, local jurisdiction(data link상의 주소: 다음 스위치와 포트)이다.

  8.3.2 Virtual-Circuit N/W source와 des.는 해당 nw또는 international nw에서 유일한 주소를 가진다. 이 global addr는 virtual-circuit identifier라는 걸 만들기 위해 사용된다.
  virtualcircuit identifier(VCI)는 switch scope(switch사이에서만 통용)을 가진다. 두

스위치 간 frame에 사용되는 것이다. 스위치에 도착하면 header안에 있는 vci가 제거됐다가 나갈때 새로운 vci로 바뀐다. source에서 des.까지 가상 회선을 만든다 = table을 채워넣는다.

> three phases

1. setup:
   1. source A에서는 근처 스위치를 통해 setup frame을 만든다.
   2. 위에서 나온 스위치는 sr frame을 받고 routing table에 따라 이동한다. routing table은 swtiching table과 다른데, 뒤 CH 에서 배우게 된다. (N/W L에서 쓴다. 라우터가 Ip addr.로 des.를 찾는 table.)
   보내면서 switching table의 incoming port/vci를 채우고, outgoing의 port까지 채운다. vci는 ack step에서 채우게 된다.
   3. 두번째 스위치는 sr frame을 받는다. 같은 일 - sw table을 채운다. 다음 스위치 vci는 이번에도 모른다.
   4. 세번째 스위치도 마찬가지이다.
   5. des. B는 sr frame을 받고, A로부터 오는 전송에 vci를 붙여준다. (다른 목적지로부터의 전송과 구분될 것이다.)
2. Acknowledgement: ack frame을 switching table을 완성하기 위해 보낸다. 1. setup때의 반대로, 세번째 스위치에 ack를 보낸다. ack에는 global addr(source, des.모두)가 들어있어서 스위치가 어느 entry가 끝났는지 (setup- sw. table을 마저 작성) 알게 한다. setup 마지막에 설정한 des. B의 vci도 넣어준다. 스위치는 sw. table의 outgoing 중 vci항목을 마저 채우게 된다. 2. 두번재, 첫번째 스위치도 마찬가지로 sw table의 outgoing - vci를 채우게 된다. 3. 첫번째 스위치가 마지막으로 ack frame을 source A에 보낸다. 3. data transfer:
source에서 des.까지 가기 위해 모든 swtich들은 virtual circuit에 관한 table을 갖고 있어야 한다.
table내용은 다음과 같다.

frame이 도착하면 스위치는 incoming 항목에서 port와 vci를 확인한다. 그후 outgoing에서 대응되는 port로 보내주며 vci를 header에 붙여준다.
이 phase는 source가 모든 frame을보낼 때까지 반복한다.

	Incoming		**Outgoing**
port	vci	port	vci
1	14	3	22

- 추가
  - teardown
    - circuit sw. nw처럼 모든 패킷은 결정된 link를 따라 전송된다.
    - source A는 des.B에게 teardown req.라는 frame을 주고 받는다.
    - B도 teardown confirmation frame을 보낸다.
    - 스위치들은 해당 virtual link에 대한 table의 entry들을 지운다.

- efficiency:
  같은 source, des.가진 패킷들은 같은 길로 간다. 패킷마다 딜레이가 다를 수도 있다.(같을 수도 있고)
  가장 큰 장점은 사용할 링크의 포화상태를 미리 확인할 수 있다는 점이다.

- delay in virtual circuit NW:
  setup과 teardown에만 있고, 스위치에서 테이블을 찾아보는 processing time이 있을 수 있다.
  스위치 안에서, physical L process - DLL process를 각각 거친 후, 바로 내보내지 않고, Queue를 통해서 나간다.
  따라서 queuing delay도 있을 수 있다.

CH.9 Introduction to Data-Link Layer

DLL에서 data frame을 physic. L로 보내면, physic. L은 data를 sig로 바꿔 전송한다.

9.1 Introduction The Internet은 라우터와 스위치 device들에 의해 이어진 NW들의 모임이다.
만일 패킷이 host에서 host로(end to end)가는 동안, 이런 NW들을 지나쳐야 할 필요가 있다.

- 서로 다른 NW는 라우터로 이어져있다.

이번 절에서는 그 중에서도 data-link L에서 통신에 관심을 가질 것이다.

- F9.1을 보면, 같은 DLL이지만, source는 하나의 NW에서 다른 host들과 route에 이어져있다.
  그러나, 어떤특정 NW로 가는 라우터의 link는 오직 하나이다.

- 9.1.1 Nodes and Links

DLL에서의 communication은 node-to-node이다.

> Internet(TCP/IP protocol을 쓰는 하나의 거대한 NW)에서의(속한) 많은 NW를(LAN, WAN) data는 지나가게 된다.
> LAN이나 WAN들은 라우터에의해 이어져있다. 이런 end host들과 router들은 `node`라고 부르고, 중간을 잇는 NW들을 `link`라고 부른다.

- 9.1.2 Services

DLL은 physical L의 service를 받아 NW L로 서비스해주는 역할을 한다.

- service의 scope은 어디까지나 node to node인 것을 기억하자.
  DLL에서, node(router나 host)들은 Internet을 떠돌아다니는 패킷을 길따라 다음 노드로 전해줘야한다.
  이를 위해, 받은 data를 encapsulate하고, 받을때는 decapsulate한다.

> source나 des.는 encapsulate, decapsulate만 각각 하지만, 중간 노드들은 둘다한다.
> 왜 중간노드들은 둘다하냐면, 중간노드는 서로 다른 protocol이라 frame format이 다를 수 있기 때문이다.
> LL addr.가 다르기 때문이기도 하다.

- Framing:
  각 노드들은 NW L로부터 받은 패킷을 보내기전에 encapsulate할 필요가 있다. 또, 받은 패킷을 decapsulate할 필요도 있다.

  > - 앞으로 나올 CH에서 배우겠지만, header랑 trailer도 있다. protocol마다 frame format차이가 있다ㅏ.

- Flow control:
  frame전송 rate와 받을 때 process rate가 불일치하면 문제가 생겨, 수신측에선 buffer가 필요하다.
  만일 버퍼가 넘치면 수신측에서 송신측에 drop rate/ stop 등의 신호를 줄 수 있다. 이 역시
  protocol마다 다르다.

- Error control:
  detection / correction

-Congestion control:
대기중인 패킷이 너무 많을 때 congestion이 발생을 감지하면, 패킷양을 sender가 줄이는 처리.
그러나 `보통 transport L에서 담당한다.`

- 9.1.3 Two Categories of Links

실제론 두 노드간 air나 cable로 physic. L에서 연결돼있지만,
이 연결(link)을 어떻게 사용할 것인지는 DLL에서 정한다.

- DLL에서 medium의 전체 capacity를 쓸 때: point-to-point link
- DLL에서 medium의 부분 capacity를 쓸 때: broadcast link

  > 다른 여러 디바이스간 공유되는 link
  > air의 경우, cellular phone쓰는 다른 많은 유저들과 공유.

  > 두 디바이스간 dedicated link

- 9.1.4 Two Sublayers

  > point-to-point, broadcast link를 다룰 때. framing, flow control 등...

1. data link control (DLC):

DLL의 service 이해를 돕기 위해, 두가지 sublayers로 나눌 수 있다.
2. media access control (MAC)

> broadcast link only. 여러 호스트들이 사용하다보니, 충돌이 발생하지 않게 해준다.

9.2 Link-Layer Addressing

CH18은 가야지 IP addr라는, NW L에서 쓰는 identifier를 배울 것이다.
이때, IP addr.는 두 종단 호스트들을 define하지만, datagram(NW에서의 패킷)들이 어디를 지나가야할지는 알려주지 못한다.

> datagram의 source IP addr.나 des. IP addr.는 바뀌지 않는다.
> 바뀌면 길을 잃거나, source가 받아야할 response, 혹은 error report를 받지 못한다.

이 경우, 두 노드간 DLL addr.이 필요하다.
DLL addr.란 link addr, physical addr. 혹은 MAC addr로 불리기도 한다.
link는 DLL에서 control되므로, DLL adddr.도 DLL에서 속해야 한다.

datagram이 전송되는 과정을 생각해보자. source에서 보낼때, datagram의 header에 source와 des.의 DLL addr가 들어간다. 이 LL addr는 매번 링크마다 바뀐다.

- 질의응답

1. source에서 des.로 가는 길에 datagram들은 IP addr.와 DLL addr를 실고 간다.

> IP addr는 source addr - des. addr순이고, DLL addr는 Des addr - source addr순으로 패킷에 들어있다.

2. router가 왜 IP주소가 필요하는지?

> CH20, 21에서 배우지만, 라우터도 message를 send하고 receive할 수 있다. 이런 프로토콜에서 필요하다.

3. 각 링크마다 router가 IP addr를 가지는 이유?

> 각각의 NW에 이어져있기 때문이다. 예를 들어 코너의 집과 같다. 같은 집이지만, 코너로 각각 난 문은 두 개의 도로명 주소를 가질 수 있다.

4. source와 des.의 IP addr는 어떻게 찾아?

> host는 본인의 IP addr를 알고 있어야 한다. CH26에 배우겠지만, app L는 DNS라는 서비스를 이용해 des. addr를 찾은 뒤, NW L로 넘겨준다.

5. 각각의 링크마다 source와 des.의 DLL addr는 어떻게 찾아?

> 각 hop(host and router)은 자신의 DLL addr.를 알고있어야 한다. des.의 DLL addr(혹은 LL addr)는 ARP라는 프로토콜을 통해 찾는다.

- 9.2.1 Three Types of addresses

같은 DLL이지만 특정 비트를 바탕으로 addr 종류를 표시한다. 예시로는 48bits (16진수로 12개의 수)를 사용하는 주소체계를 보인다.

> 1. Unicsat Address:
>   1-1 통신에서 사용한다. 하나의 링크를 사용하는 host와 router들 사이에는 unicast addr.가 할당돼있다.
>   2번째수가 odd.
> 2. Multicast address:
>   하나의 source를 다수의 이용자가 사용하는 통신에서 쓴다. 하지만, local scope가 있다.
>   2번째수가 even.
> 3. broadcast addresss:
>   some LL protocol들은 broadcast addr.를 정해놨다. LAN같은 경우, 특히!
>   모든 bit가 1이다.

- 9.2.2 ARP
  link따라 다른 노드로 IP datagram을 보낼 때, des.의 IP addr.를 포함한다.
  host는 default router의 Ip addr.를 알고 있다.
  각각의 라우터는 마지막 router를 제외하고는 next router의 IP addr를 forwarding table을 이용해 알아내야 한다.

이때, IP addr는 link상에서 움직일 때 도움이 안된다.
next node의 DLL addr가 필요하다. 이때 ARP를 쓴다.

```
 - ARP는 NW L에서 정의된 프로토콜이다. IP addr.를 보고 DLL addr를 mapping해준다.
```

1. DLL addr가 필요하면, 받을 노드의 IP addr, sender의 IP addr와 DLL addr를 포함해 ARP request packet을 보낸다.
이때, DLL addr를 몰랐기 때문에, 쿼리(요청)은 broadcast된다.
2. 본인 IP가 맞는 receiver는 본인 IP addr.와 DLL addr를 담아 response packet을 unicast로 보낸다. (물론, 받아야할 node의 IP, LL addr도 넣었을 것이다.)

ARP packet format

Hardware type(16bit)		**Protocol Type(16bit)**
HW length	Protocol length	operation request:1, reply:2 (flag)
		source hardward addr
		source protocol addr
		des. HW addr(LLD)(empty in request)

Hardware type(16bit)	Protocol Type(16bit)
	des. protocol addr(IP)

- 9.2.3 An Example of Communication
source A가 des. B로 통신을 하고 싶다.

1. data가 NW L로 내려온다. A(자신) IP addr와 B IP addr는 알고 있다.
2. datagram을 만든 후(header에 A,B IP addr를 넣는다.) DLL로 내려보낸다.
3. 이때, DLL로 내려보내기 전, des.의 LL addr.를 알려줘야 한다. A는 B의 IP addr를 가지고 자신의 forwarding table을 살핀다.
4. fw table이 다음 라우터로 가는 IP를 알려준 뒤, A는 해당 라우터에서 ARP req.를 통해 (broadcast) reply(unicast)를 받아서 DLL addr를 DLL로 같이 내려 보낸다.
5. DLL에서는 header에 DLL addr(다음 라우터로 가는 DLL addr)를 넣는다.
6. 계속 내려가다 physic. L에서 전송한다.

(중간고사 범위 끝.)

In [ ]: