소개

AMM31030-03 네트워크 기초

정내훈

2021년도 1학기

한국산업기술대학교 게임공학부

내용

- 강좌 소개

- “데이터 통신”에 대한 소개

강사 소개

- 경력
  - 1990년부터 온라인게임 개발
    - LPMUD, Archmage
  - 2002년 3월 – 2008년 2월 NCSoft
    - MMORPG 개발 : Lineage forever, Alterlife, Blade & Soul
  - 2015,2016,2017 모바일 게임 서버 개발 Netmarble
    - GSF, FinalShot, Lineage2Revolution
- 전공
  - Parallel Processing
- 관심분야
  - 차세대 게임 서버 구조
- 연락처
  - nhjung@kpu.ac.kr 공학관 E동 314호

개요 – 네트워크 기초

- 목적
  - 온라인 게임개발에 필요한 네트워크에 관한 기본 이론과 응용들에 대한 지식을 습득한다.

개요 – 게임서버 프로그래밍

- 교재
  - 미즈구치 카즈야, “모두의 네트워크”, 2018, 길벗
    - 최신
    - 쉽다

강의 계획

- 1 주 : 데이터 통신 소개
- 2 주 : LAN 기초
- 3 주 : LAN 구조와 연결
- 4 주 : 네트워크 운영체제
- 5 주 : 음성 네트워크
- 6 주 : WAN과 보안
- 7 주 : 인터넷

강의 계획

- 8 주 : 중간고사
- 9 주 : TCP/IP와 소켓 프로그래밍
- 10 주 : 게임 네트워크 서버의 구성
- 11 주 : 유무선공유기와 IPV6
- 12 주 : 이동통신기술 1G-5G
- 13 주 : 가상화 및 클라우드 시스템
- 14 주 : IOT, 네트워크 보안, 개인정보 보호
- 15 주 : 학기말 고사

강의 계획

- 기존의 김재경 교수님의 강의와의 차이
  - 첫 강의이므로 시행착오들이 있을 예정
  - 주제 변경
    - 좀더 게임에 가까운 또는 강사가 익숙한 주제로
    - IPTV -> 클라우드 시스템
  - 조별 발표 생략
    - 코로나 시대를 맞이하여.
    - 대신 프로그래밍 과제
      - C++11, VisualStudio 2019

성적 산출

- 중간 고사 : 30%
- 학기말 고사 : 30%
- 과제 : 30%
- 출석 : 10%
  - 결석 1번에 1% 감점, 지각 3번에 1% 감점, ¼이상 결석 F)

선수 과목

- MM1240 : C++
  - 예제 설명 및 과제 수행에 필요

강의 편성

- 숙제
  - 강의 내용 실습
  - C++로 강의 내용을 구현해 보는 과제
  - Eclass를 통해 제출
  - Delay시 하루당 10%감점

개발 환경

- 운영 체제
  - windows 10
- 언어
  - C++11
  - Visual Studio 2019 (community)

1장

데이터 통신 – 소개

목표

- 데이터 통신과 구성 요소 정의.
- 3가지 종류의 데이터 표현
- 아날로그와 디지털 데이터의 차이.
- 아날로그 전송과 디지털 전송의 차이
- 병렬 전송과 직렬 전송의 차이

목표 (계속)

- 동기와 비동기식 전송의 차이
- 단방향 통신, 반이중 통신, 전이중 통신
- 데이터 통신 매체
- 데이터 통신 표준, 표준 기구, 표준 제정 방법
- OSI 와 TCP/IP모델의 레이어 구조

데이터 통신의 정의

- A지점에서 B지점으로 데이터를 이동시키기
- 반드시 한 개 이상의 전송 매체가 필요.
- 전송 매체를 통과하기 위해 데이터는 변환되어야 함.
  - 적절한 자료구조로 변환 되어야 함
- HW, SW 및 적절한 운영이 필요함
- 여러 노드, 사람, 업무들 사이에 매체를 통해 정해진 포맷에 데이터가 전달됨을 의미

BITS, BYTES, DATA ENCODING

- 사람이 알아볼 수 있는 데이터들은 컴퓨터가 인식할 수 있는 데이터로 변환 되어야 한다. 이 때 bits, bytes, 와 데이터 변환이 사용된다.

- Bit – 2진수에서 가장 작은 변환 단위
- Byte – 8 bits.
- 데이터 변환(Data Encoding) – 디지털 혹은 이진법으로 데이터가 표현되는 방식

BITS, BYTES, 그리고 DATA ENCODING (계속)

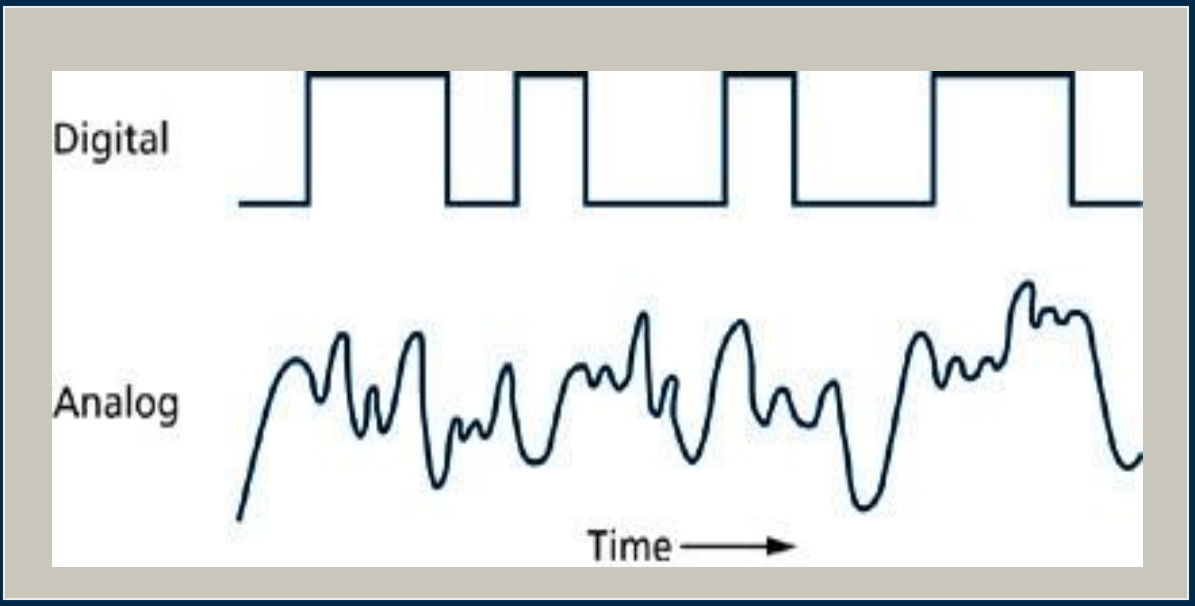
- 예:
  - EBCDIC – the Extended Binary Coded Decimal Interchange Code.
    - 옛날 IBM, 50년 전, 지금은 안 씀
  - ASCII – the American Standard Code for Information Interchange.
    - 옵션) UTF-8 예) char ch = ‘A’;
  - Unicode – surpasses the limitations of ASCII by employing more bits.
    - 한국어 표준

디지털, 아날로그

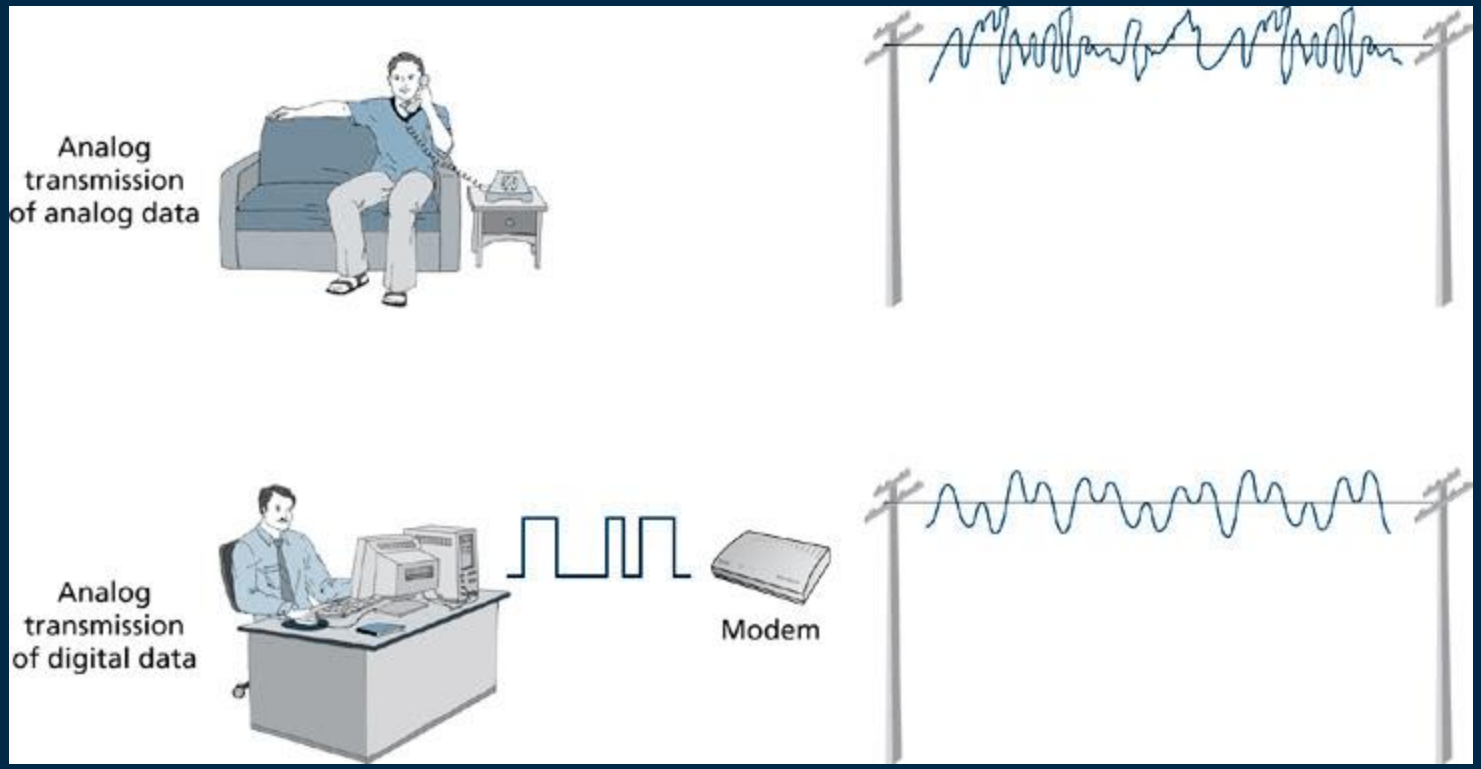
- 아날로그 데이터 – 연속적인 값을 갖는 소리, 빛, 전압 등으로 표한

- Digital data – 불연속적인 값을 같는 소리, 빛, 전압 등으로 표현

디지털 전송과 아날로그 전송

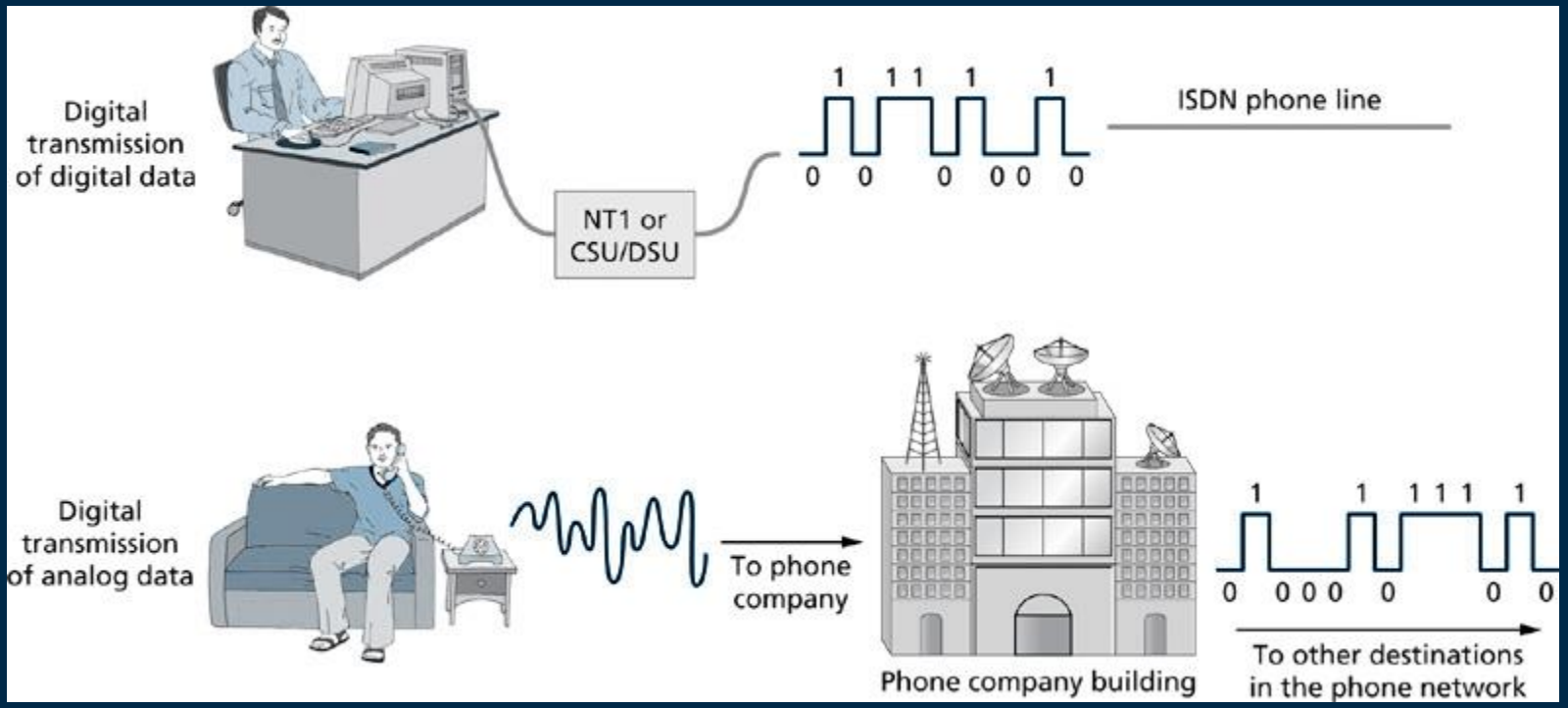


디지털 전송과 아날로그 전송



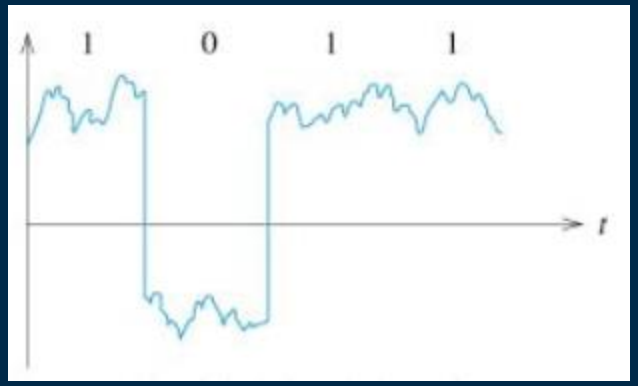
모뎀????

디지털 전송과 아날로그 전송



디지털 전송과 아날로그 전송

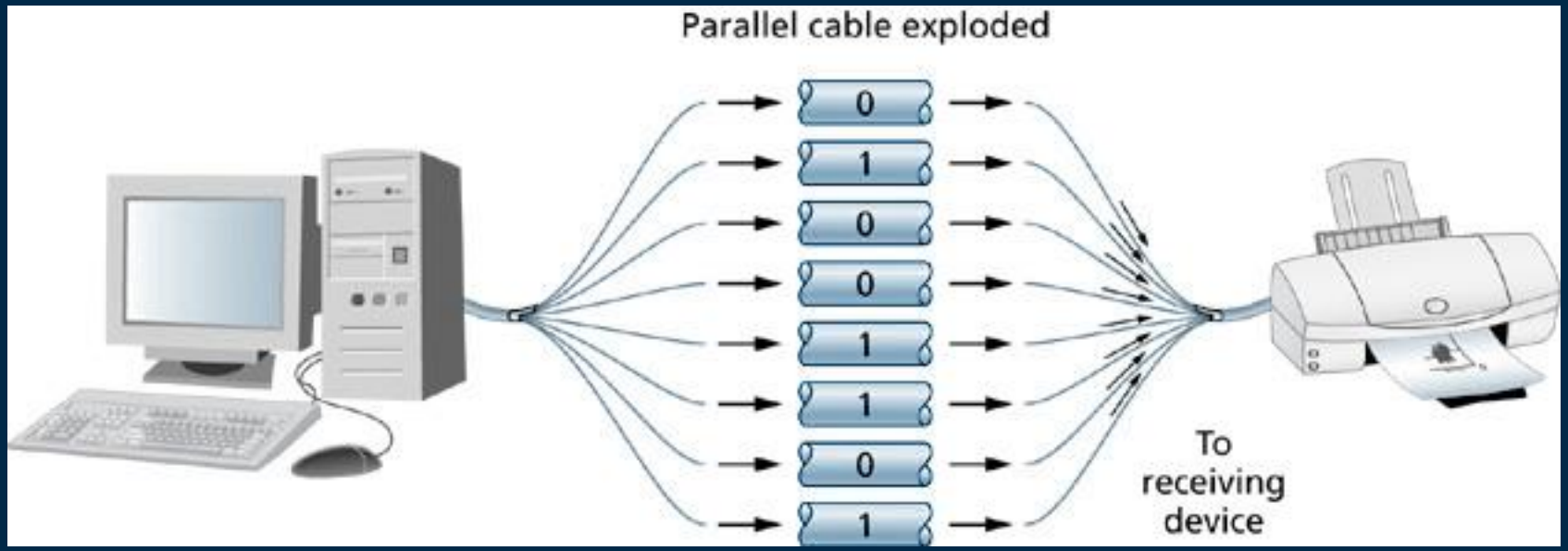
- 디지털의 구현
  - 모든 물리 데이터는 아날로그
  - 아날로그 데이터를 디지털로 정의
    - 예) -1.5V ~ +1.5V => 0, 3.5V ~ 6V => 1



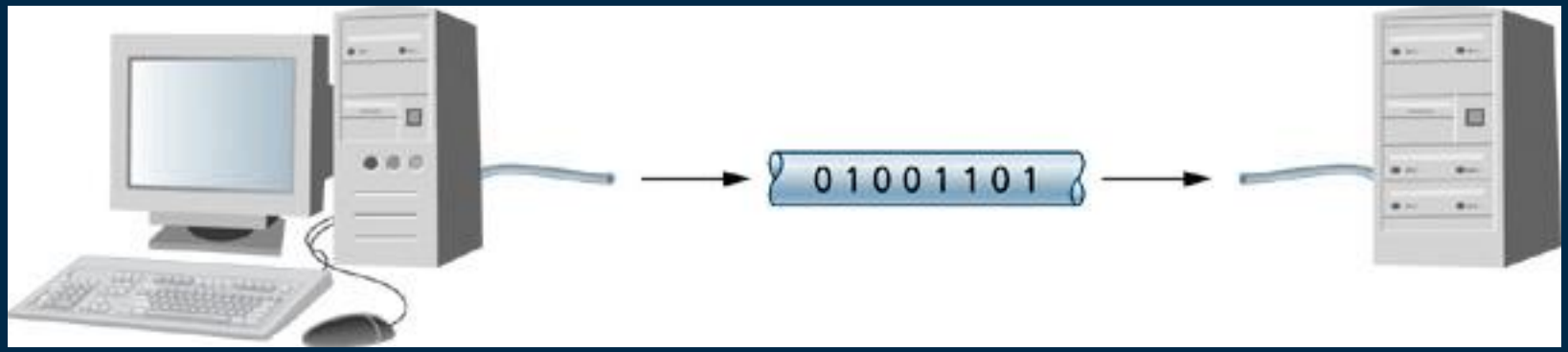
디지털 전송과 아날로그 전송

- 디지털의 장점
  - 오류에 강하다.
    - 약간의 오류는 결과에 영향을 끼치지 않는다.
    - 오류 여부를 판단 가능하다. (예: 체크섬)
- 아날로그의 장점
  - 정보량이 많다.
- 아날로그의 단점
  - 오류 (=noise)에 약하다.
  - 컴퓨터에 저장하기 위해서는 변환이 필요하다.

병렬 전송



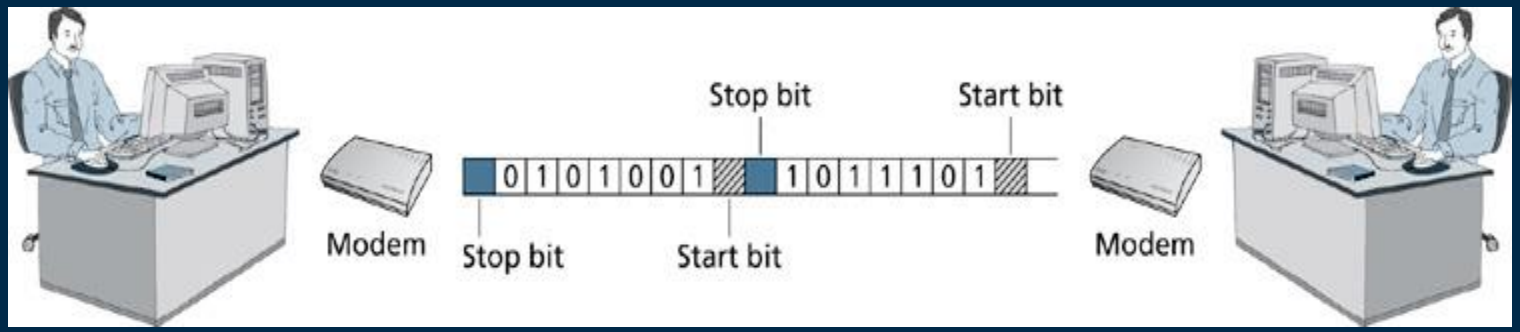
직렬 전송



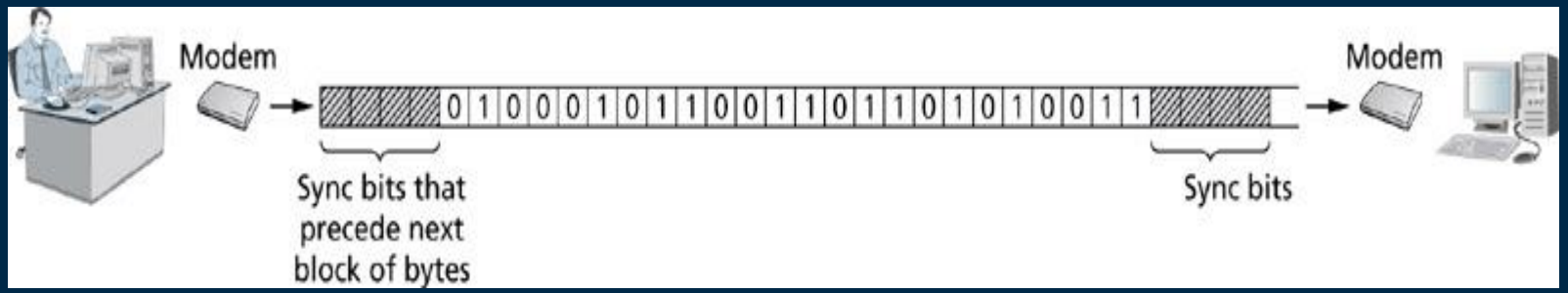
병렬 전송과 직렬 전송

- 현재는 모두 직렬전송으로 바뀌었고 병렬 전송은 극소수만 남아 있음
  - 극소수 : CPU <-> MEMORY
- 전환
  - PCI bus -> PCI Express bus
  - Parallel Port -> USB(Universal Serial Bus)
  - IDE -> SATA
- 원인
  - 전송 속도가 빨라짐으로 인해 회선 사이의 간섭 노이즈가 커지고 회선들 사이의 전송 속도를 맞추는 것이 어려워짐.

비동기(Asynchronous) 전송



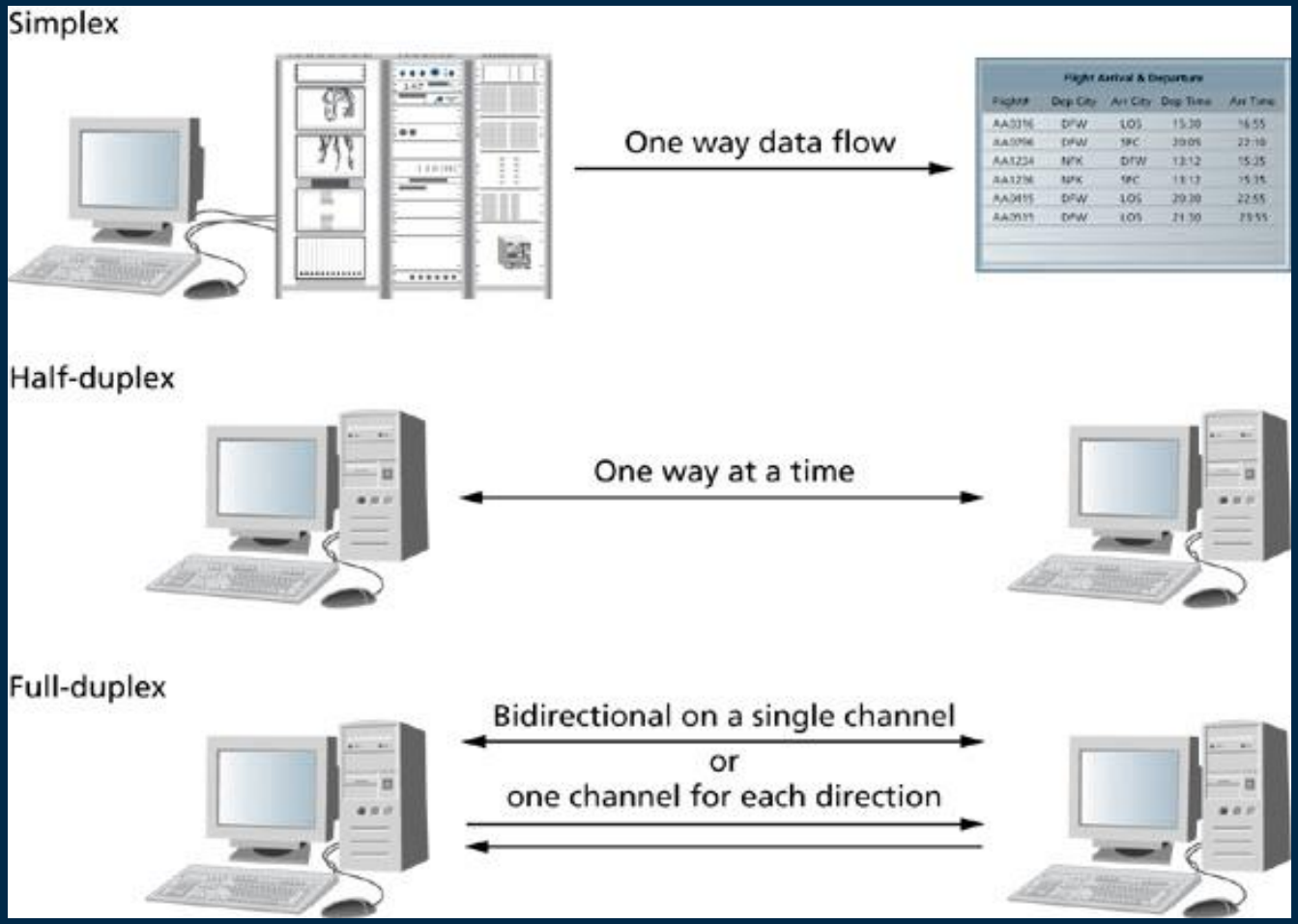
동기식(Synchronous) 전송



동기, 비동기 전송

- 동기
  - 데이터를 묶어서 전송 (대량 고속 전송에 적합)
  - 시간 단위로 비트를 구분하므로 공통 클럭이 필요 (클럭을 위한 별도 회선 필요)
  - 하드웨어가 비쌈
- 비동기
  - 데이터를 짧게 나누어 전송
  - 공통 클럭 없이 시작 신호와 종료 신호로 데이터를 구분
  - 불규칙 소량 데이터 전송에 적합
- 자세한 내용은 전자 공학과에서

단방향(Simplex), 반이중(Half-Duplex), 전이중(Full-Duplex) 전송



데이터 통신 표준

- 표준은 공인된 모델이나 규격
- 데이터 통신과 네트워크에서 표준은 널리 사용됨
  - 왜? 말이 다르면 대화가 안되니까.
- 표준은 기기간의 기본 호환성과 상호 동작을 보증
- 모스 부호나 음성 전화도 표준의 역사적인 예

데이터 통신 표준(계속)

- 많은 표준 단체에서 데이터 통신 표준을 개발하고 공표하고 있음.

- ANSI – 미국 국립 표준 협회 (American National Standards Institute) 민간단체, ISO에 가입되어 있음.
- IEEE – 전기전자공학자협회(Institute of Electrical and Electronics Engineers) 전기, 전자, 컴퓨터 기술 공유와 표준 정의

데이터 통신 표준 (계속)

- ITU – 국제전기통신연합 (International Telecommunication Union) 정부간 국제 기구로, 정보통신기술 관련 문제를 책임지는 유엔 전문 기구.
- ISO – (**I**nternational **O**rganization for **S**tandardization - https://www.iso.org) 국제 표준 기구. 데이터 통신표준도 제정한다.

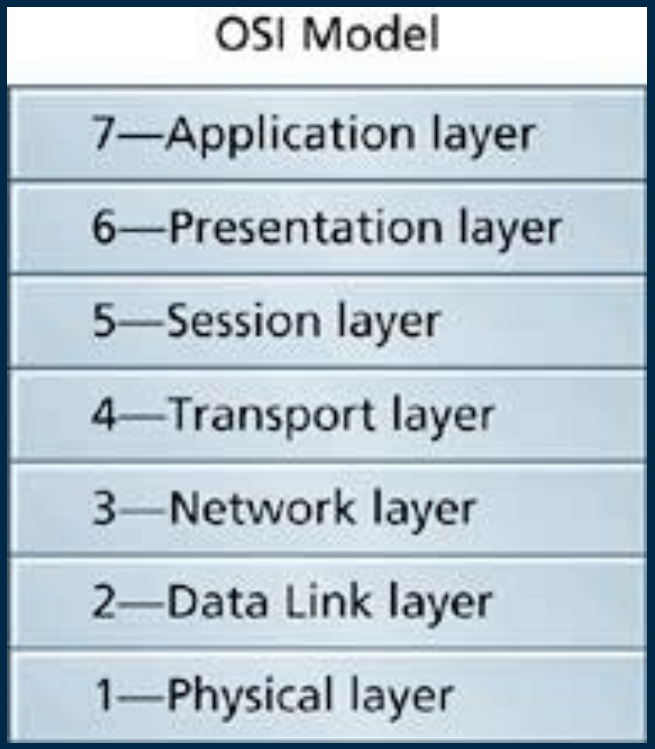
데이터 통신 모델

- 계층 구조와 **프로토콜**로 정의되는 2개의 중요한 모델이 있다.
  - OSI model
  - TCP/IP model.
- 프로토콜 : 주고 받는 데이터의 포맷과 전송 순서의 정의
- 지금의 인터넷은 TCP/IP모델로 구현되어 있으며, 인터넷을 이해하기 위해서는 이 모델을 숙지해야 한다.
- OSI모델은 개념 설명을 위한 모델로 사용되는 곳이 없다.
  - 레이어가 너무 많다.

OSI 모델

- 1970년대 후반에 정의.
- 7-계층으로 각 계층은 기능 및 주고 받는 프로토콜로 정의 되어있다.
  - 계층단위로 기능을 정의하므로 복잡한 구조를 여러 개의 간단한 구조로 표현
  - 같은 계층 끼리의 프로토콜과 바로 위/바로 아래 계층과의 프로토콜의 정의로 충분
    - 이것 이외의 다른 연결을 신경 쓸 필요가 없어 져서 정의 및 구현이 단순해 짐

OSI 계층 모델


OSI Model
7—Application layer
6—Presentation layer
5—Session layer
4—Transport layer
3—Network layer
2—Data Link layer
1—Physical layer

OSI 계층 모델

- *물리계층 Physical Layer*
  - OSI모델 의 1계층
  - 디지털 데이터를 물리적 신호로 변환
  - 기기간의 물리적 연결 및 Bit의 전달을 위한 프로토콜을 정의
    - 예) 소켓 규격, 신호의 강도, 신호의 시작과 끝
    - 디지털? 아날로그?
    - 동기? 비동기?
    - 단방향? 전 이중?
  - 전송 속도를 정한다. 10 Mbps, 100 Mbps, 1000 Mbps, etc.

OSI 계층 모델

- *데이터 링크 계층 Data Link Layer*
  - 포인트 to 포인트 사이의 신뢰성 있는 전송 보장
  - bit들의 모임인 frame을 구성
    - frame : 한번에 주고 받는 데이터 단위, low-level packet
  - 단말(노드) 주소를 갖고 있다.
    - 예) MAC 주소
  - 오류 탐지와 수정 기능을 갖는다.

OSI 계층 모델

- *네트워크 계층 Network Layer*
  - 논리 네트워크와 논리 주소를 정의
  - 데이터를 주소가 포함된 패킷들로 분할하여 전송
    - 패킷
      - 네트워크를 통해 이동하는 데이터의 묶음
      - 사람들이 데이터라면 그들이 탄 비행기 한대가 패킷
      - 전체가 전달되거나 하나도 전달되지 않거나 둘 중의 하나.
      - Frame과는 다르다.
  - 패킷의 주소를 참조하여 최적의 경로를 판단 하여 전송

OSI 계층 모델

- *전송 계층 Transport Layer*
  - 사용자 사이의 신뢰성 있는 데이터 전송을 구현
    - 사용자 => 프로그램
  - 연결 기반(Connection-oriented) 데이터 전송 제공
  - 에러 검사, 재전송, 패킷 순서 정렬을 통해 신뢰성 구현
    - 데이터가 보낸 순서대로 에러 없이 도착
  - 서비스 주소 (포트 번호) 정의
    - 네트워크 주소는 컴퓨터 주소
    - 서비스 주소는 프로그램 선택 용 번호 (기능 별 번호 부여)

OSI 계층 모델

- *세션 계층 Session Layer*
  - 연결을 만들고, 관리하며, 동기화하고, 종료하는 작업을 하는 계층

OSI 계층 모델

- *표현 계층 Presentation Layer*
  - 데이터 변환 서비스 담당
    - 예) 변환 – ASCII, EBCDIC, or Unicode.
  - 전송을 위한 암호화와 복호화
  - 압축 및 해제

OSI 계층 모델

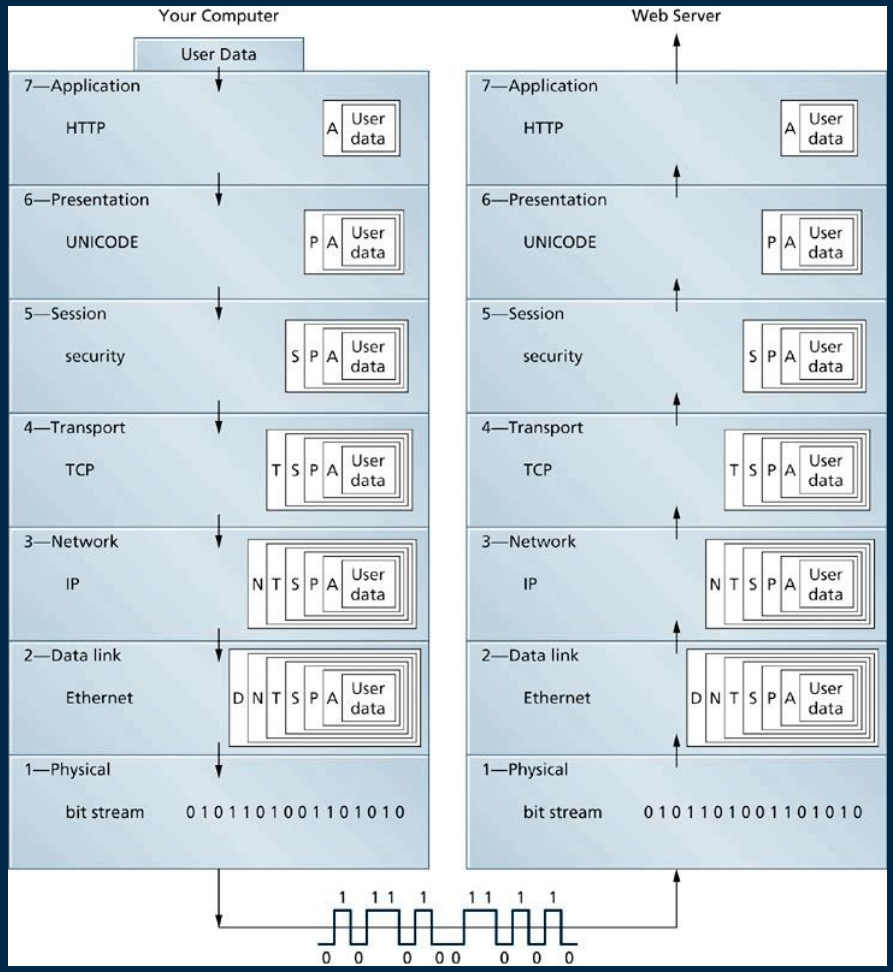
- *응용 계층 Application Layer*
  - 실제 프로그램과 관계하여 응용 서비스를 담당. 파일, 프린터, 이메일 등의 서비스를 제공
    - 예) % rcp userB@node.a:file_c | cat /dev/printer1
  - 네트워크를 사용하는 프로그램이 속한 계층
    - 예) 온라인 게임 서버, 온라인 게임 클라이언트

계층 구조에서의 데이터 캡슐화

- 데이터 캡슐화(Data encapsulation) 는 기존데이터에 헤더와 같은 프로토콜 정보를 추가해서 상대측 같은 레이어에 보내는 데이터 묶음(패킷)을 만드는 과정이다.

- 데이터를 보낼 때는 상대방의 같은 레이어가 받을 데이터만 신경 써서 작성하면 되며, 보낼 때는 아래 레이어에 전달하면 된다. 데이터를 받았다면 캡슐화된 데이터를 원상 복귀한 후 위의 레이어로 보내면 된다.

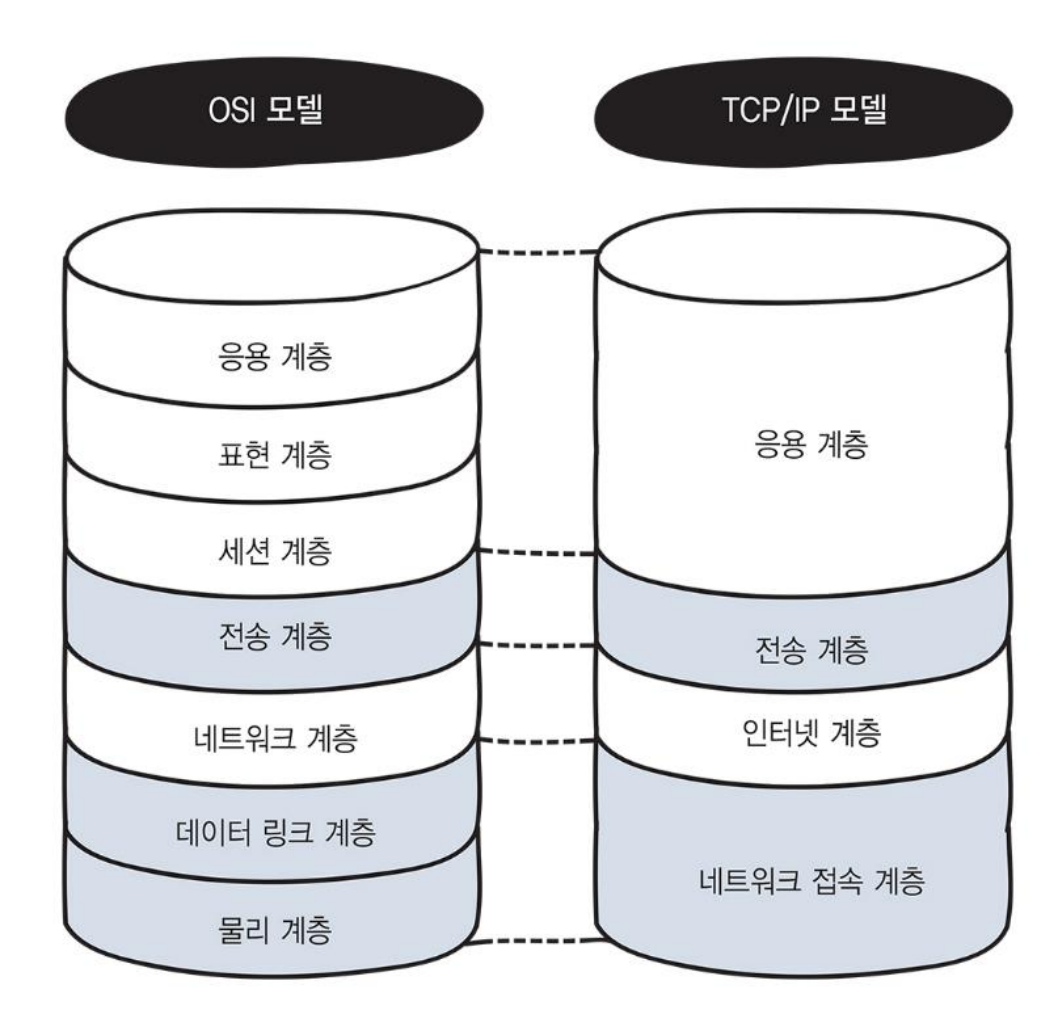
계층 구조에서의 데이터 캡슐화



TCP/IP 모델

- 1970년대 초부터 개발 시작
- 4개의 계층으로 이루어 짐
  - 모호한 계층이 없고, 레이어 전달 Overhead 최소화
- 현대의 인터넷을 구성

TCP/IP 모델



교재 88 페이지

TCP/IP 모델

중구난방 용어 정리

RFC 1122, Internet STD 3 (1989)	Cisco Academy[36]	Kurose,[37] Forouzan[38]	Comer,[39] Kozierok[40]	Stallings[41]	Tanenbaum[42]	Arpanet Reference Model (RFC 871)	OSI model
Four layers	*Four layers*	*Five layers*	*Four+one layers*	*Five layers*	*Five layers*	*Three layers*	*Seven layers*
"Internet model"	"Internet model"	"Five-layer Internet model" or "TCP/IP protocol suite"	"TCP/IP 5-layer reference model"	"TCP/IP model"	"TCP/IP 5-layer reference model"	"Arpanet reference model"	OSI model
Application	Application	Application	Application	Application	Application	Application/Process	Application
							Presentation
							Session
Transport	Transport	Transport	Transport	Host-to-host or transport	Transport	Host-to-host	Transport
Internet	Internetwork	Network	Internet	Internet	Internet		Network
Link	Network interface	Data link	Data link (Network interface)	Network access	Data link	Network interface	Data link
		Physical	(Hardware)	Physical	Physical		Physical

Internet protocol suite - Wikipedia

TCP/IP 모델 – 계층 1

- 네트워크 접속 계층 (Network Access Layer) 링크 계층(Link Layer)
- 디바이스 끼리의 1 대 1 통신을 제공
  - HW 정의 포함
  - MAC 주소로 통신 대상 지정
- 다양한 매체 지원
  - 광케이블, 구리전선, 전파, 적외선등
  - 자세한 내용은 다음장에서

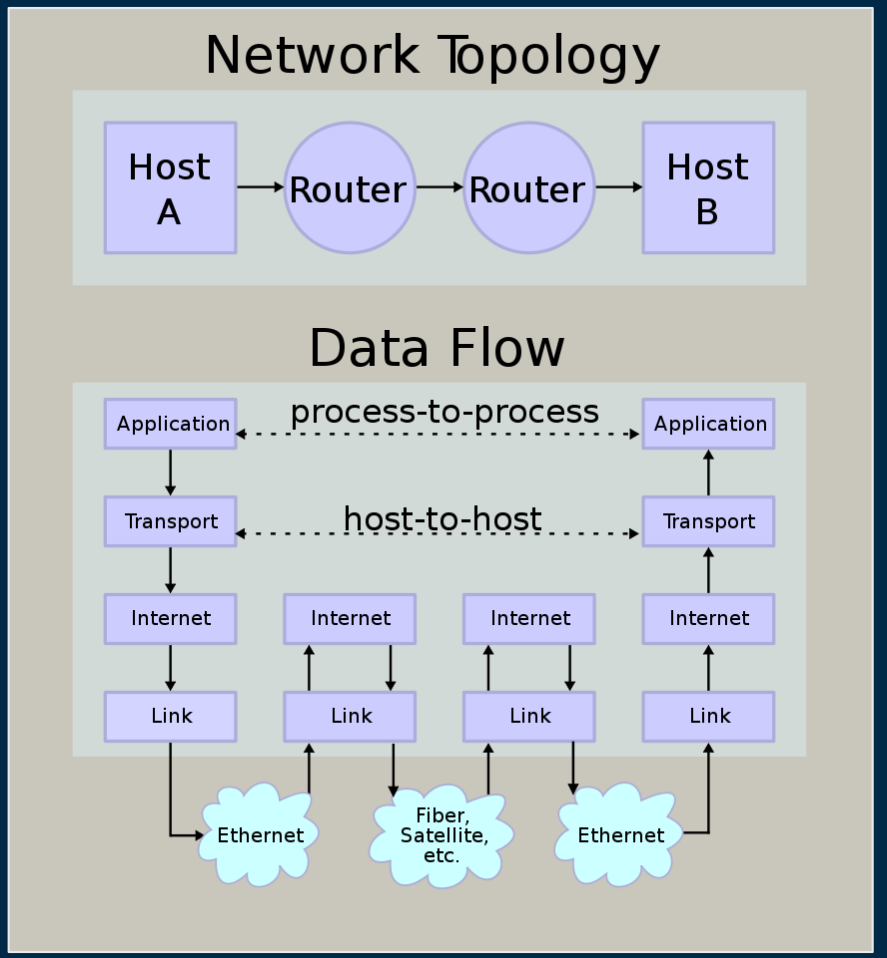
TCP/IP 모델 – 계층 2

- 인터넷 계층 (Internet Layer)
- 라우팅 기능 제공
  - 라우팅 : 패킷을 받았을 때 자신이 수신자가 아닐 경우 목적지에 가까운 상대방에게 전달해 주는 기능
  - IP 주소로 목적지 선택

TCP/IP 모델 – 계층 2

- 라우팅
  - 데이터의 출발지와 목적지가 바로 연결되어 있는 경우는 거의 없음
    - 예외) 개발 단계
  - 중간에 기지국, 공유기 등을 통과 함.
    - 택배 물류 센터를 생각하면 됨

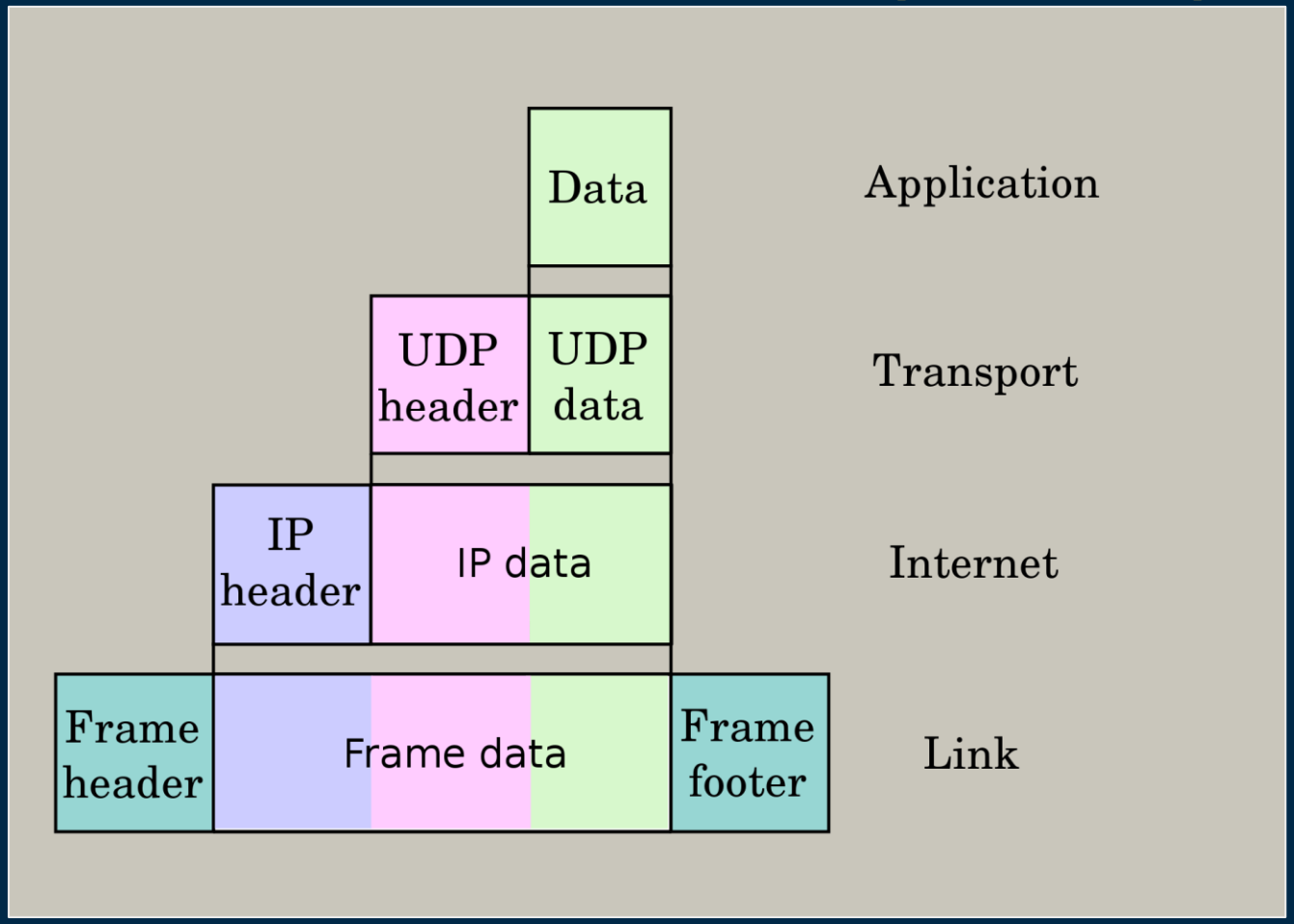
Internet protocol suite - Wikipedia



TCP/IP 모델

- 데이터 캡슐화

Internet protocol suite - Wikipedia



TCP/IP 모델 – 계층 3

- 전송 계층 (Transport Layer)
- 프로세스 사이의 1 대 1 전송 기능 제공
- IP 주소는 컴퓨터만 구분
  - Port 번호로 프로세스 구분
  - 예) 집 주소, 사람 이름
- 송신 프로세스가 보내는 데이터를 그대로 수신 프로세스에 전달.
  - 순서 맞추기, 오류 검사, 재전송

TCP/IP 모델 – 계층 3

- 두가지 모드 제공
  - TCP (Transmission Control Protocol)
    - Reliable Byte Stream 제공
    - 순서, 오류 검사 기능, 재전송을 통한 완전한 데이터 전송
  - UDP (User Datagram Protocol)
    - 순서, 오류 검사, 재전송 없음
      - 프로그램의 책임
    - TCP에 비해 낮은 오버헤드, 빠른 전송
    - 음성, 영상 등에 사용. 게임에서는 덜 중요한 데이터에 사용

TCP/IP 모델 – 계층 4

- 응용 계층 (Application Layer)
- 네트워크 자체의 구현이 목적이 아닌 네트워크를 사용해 실제로 유용한 작업을 하는 계층
  - 예) 게임서버, 게임클라이언트, 웹서버, 웹브라우저, 카카오톡
- 우리가 작성하는 모든 네트워크 프로그램.

숙제 1 (1/5)

- TCP/IP 모델의 링크 계층 구현하기
- 두개의 컴퓨터 노드 A와 노드 B는 하나의 전선 g_conn으로 연결되어 있다. 각각의 노드는 g_conn을 true또는 false로 변경할 수 있고, 변경하면 상대 노드에도 변경한 값이 보인다.
- 이 때 노드 A에서 노드 B로 문자를 전송하도록 노드 A와 노드 B를 구현하라.

숙제 1 (2/5)

- 구현은 NODE_A 솔루션의 do_node_a()함수와 NODE_B 솔루션의 do_node_b()함수를 변경해서 구현하시오
  - 다른 함수들은 수정하시 마시오.
  - 숙제 검사는 do_node_a()함수와 do_node_b()함수를 복사하여 검사할 예정임
  - 표준 라이브러리 사용을 위한 #include의 추가는 가능
- NODE_A와 NODE_B솔루션은 e_class에 올려져 있음.
  - g_conn의 값을 테스트하는 샘플이 구현되어 있음.

숙제 1 (3/5)

- 힌트
  - 데이터를 보낼 때 시작 신호를 보낸 후 데이터를 비트들로 쪼개어 일정 시간마다 보내면 된다.
  - 일정 시간 간격 동안 기다리는 함수는 std::this_thread::sleep_for(시간)가 있다.

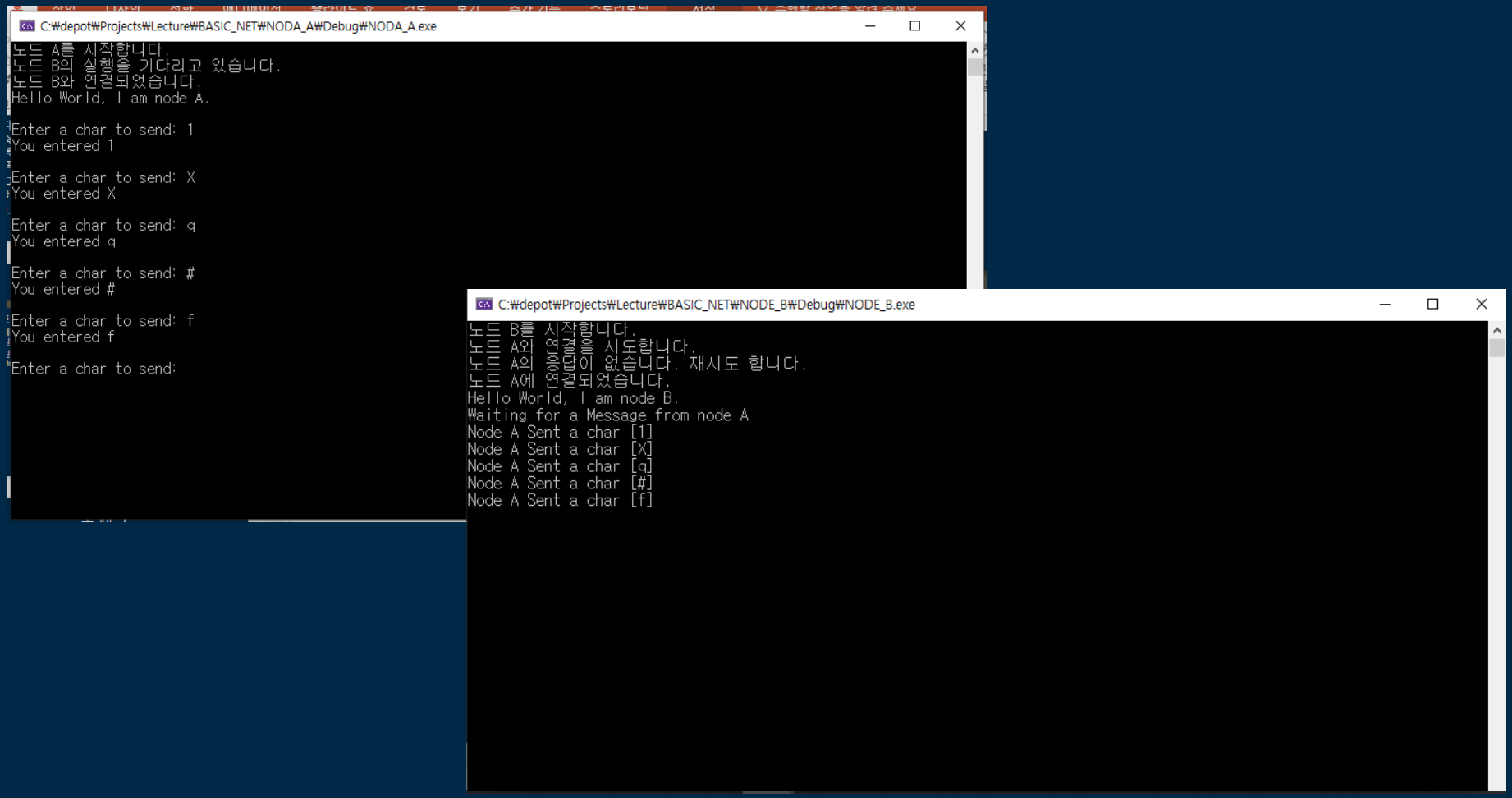
```
#include <iostream>
#include <thread>

using namespace std;

int main()
{
   cout << "Start\n";
   this_thread::sleep_for(100ms);
   cout << "After 100ms\n";
}
```

숙제 1 (4/5)

- 제출 내용
  - node_a.cpp와 node_b.cpp 파일
  - 구현한 방법에 대한 설명
  - 실행 스크린 샷
- 제출 방법
  - eclass에 제출

숙제 1 (5/5)

- 구현 예


C:₩depot₩Projects₩Lecture₩BASIC_NET₩NODA_A₩Debug₩NODA_A.exe
노드 A를 시작합니다.
노드 B의 실행을 기다리고 있습니다.
노드 B와 연결되었습니다.
Hello World, I am node A.
Enter a char to send: 1
You entered 1
Enter a char to send: X
You entered X
Enter a char to send: q
You entered q
Enter a char to send: #
You entered #
Enter a char to send: f
You entered f
Enter a char to send:
C:₩depot₩Projects₩Lecture₩BASIC_NET₩NODE_B₩Debug₩NODE_B.exe
노드 B를 시작합니다.
노드 A와 연결을 시도합니다.
노드 A의 응답이 없습니다. 재시도 합니다.
노드 A에 연결되었습니다.
Hello World, I am node B.
Waiting for a Message from node A
Node A Sent a char [1]
Node A Sent a char [X]
Node A Sent a char [q]
Node A Sent a char [#]
Node A Sent a char [f]