Socket flags Option

- MSG_OOB : Out-of-band 형태로 데이터가 전송되려면 별도의 통신 경로가 확보되어서 고속으로 데이터가 전달되어야 하지만 TCP에서는 지원하지 않아 전송순서는 유지하되 앞의 데이터 처리를 신속하게 할 것을 요구한다.

fcntl(recv_sock, F_SETOWN, getpid( )) : recv_sock에 의해 발생하는 SIGURG 시그널을 처리하는 프로세스를 getpid 함수가 반환하는 ID의 프로세스로 변경시킨다.

- MSG_DONTWAIT : Non-blocking IO의 요구

- MSG_PEEK 옵션 : MSG_DONTWAIT 옵션과 함께 설정되어 입력버퍼에 수신 된 데이터가 존재하는지 확인하는 용도

  * recv함수를 호출하면 입력버퍼에 존재하는 데이터가 읽혀지더라도 입력버퍼에서 데이터가 지워지지 않는다.

readv & writev IO function

- 데이터 송수신의 효율성을 향상시키는데 도움이 되는 함수
- 데이터를 모아서 전송하고, 모아서 수신하는 기능의 함수

writev : 여러 버퍼에 나뉘어 저장되어 있는 데이터를 한번에 전송

```
#include  < sys/uio.h >

ssize_t  writev( int filedes, const struct iovec * iov, int iovcnt )
```

- filedes : 데이터 전송의 목적지 소켓의 파일 디스크립터 / 파일이나 콘솔도 대상이 될 수 있다.
- iov : 구조체 iovec 배열의 주소 값
- iovcnt : iov가 가리키는 배열의 길이정보 전달

```
struct iovec {
    void *  iov_base;  // 버퍼의 주소 정보
    size_t iov_len;    // 버퍼의 크기 정보
}
```

readv : 데이터를 여러 버퍼에 나눠서 수신

```
#include  < sys/uio.h >

ssize_t  readv( int filedes, const struct iovec * iov, int iovcnt )
```

- filedes : 데이터 수신할 파일 / 소켓의 파일 디스크립터
- iov : 데이터를 저장할 위치와 크기 정보를 담고 있는 iovec 구조체 배열의 주소 값
- iovcnt : iov가 가리키는 배열의 길이정보 전달

Multicast

- 멀티캐스트 방식의 데이터 전송은 UDP를 기반으로 하며 UDP 서버/클라이언트의 구현방식이 매우 유사하다.

- 단 한번에 데이터 전송으로 다수의 호스트에게 데이터를 전송할 수 있다.

- 특성
  1. 멀티캐스트 서버는 특정 멀티캐스트 그룹을 대상으로 데이터를 딱 한번 전송한다.
  2. 멀티캐스트 그룹의 수는 IP주소 범위 내에서 얼마든지 추가가 가능하다.
  3. 특정 멀티캐스트 그룹으로 전송되는 데이터를 수신하려면 해당 그룹에 가입하면 된다.

- 다수의 UDP 패킷을 전송하는 것이 아닌 하나의 패킷을 네트워크상에서 라우터들이 복사하여 다수의 호스트에게 전달

- 멀티미디어 데이터의 실시간 전송 : 하나의 영역에 동일한 패킷이 둘 이상 전송되지 않는다.

- 적지 않은 수의 라우터가 멀티캐스트를 지원하지 않기 때문에 터널링 (Tunneling) 기술을 사용

터널링 (Tunneling)
- 한 네트워크에서 다른 네트워크의 접속을 거쳐 데이터를 전송하는 기술 - 두 번째 네트워크에 의해 운송되는 패킷들 내에 네트워크 프로토콜을 캡슐화함으로써 운영 - 멀티캐스트 주소를 가진 패킷 헤더 앞에 멀티캐스트 라우터간에 설정된 터널의 양 끝단의 IP를 덧붙여 라우팅을 함으로써 멀티캐스트를 지원하지 않는 라우터를 거칠 때 유니캐스트 패킷과 같은 방법으로 리 이루어 지도록 한다.

- 멀티캐스트 패킷의 전송을 위해서는 '패킷을 얼마나 멀리 전달할 것인가'를 나타내는 TTL을 반드시 설정해야 한다.

TTL (Time to Live) 설정 방법
- 프로그램상에서의 TTL 설정은 소켓의 옵션설정을 통해 이루어 진다. (setsockopt 함수) - TTL의 설정과 관련된 프로토콜의 레벨은 IPPROTO_IP이며 옵션의 이름은 IP_MULTICAST_TTL이다. - ex) setsockopt(send_sock, IPPROTO_IP, IP_MULTICAST_TTL, (void*)&time_live, sizeof(time_live))

멀티캐스트 그룹으로의 가입
- 소켓의 옵션설정을 통해 이루어진다. (setsockopt 함수) - 프로토콜의 레벨은 IPPROTO_IP이고, 옵션의 이름은 IP_ADD_MEMBERSHIP이다. - struct ip_mreq 구조체를 사용한다. - ex) setsockopt(recv_sock, IPPROTO_IP, IP_ADD..., (void*)&join_adr, sizeof(join_adr));

```
struct ip_mreq {
    struct in_addr imr_multiaddr;      // 멀티캐스트 그룹의 주소 정보
    struct in_addr imr_interface;      // 그룹에 가입할 호스트의 주소 정보
}
```

Broadcast

- 한번에 여러 호스트에게 데이터를 전송한다는 점에서 멀티캐스트와 유사하지만 전송이 이뤄지는 범위에서 차이가 있다.
  (멀티캐스트는 다른 네트워크상에 존재하는 호스트라 할지라도, 멀티캐스트 그룹에 가입만 되어 있으면 데이터의 수신이 가능하지만 브로드캐스트는 동일한 네트워크로 연결되어 있는 호스트만 제한된다.)

- Directed Boradcast VS Local Broadcast
  -> Directed Boradcast : 네트워크 주소를 제외한 나머지 호스트 주소를 전부 1로 설정 [ex) 192.12.34.255]
  -> Local Broadcast : 255.255.255.255라는 IP주소가 특별히 예약되어 있다. = 해당 네트워크 모든 호스트에게 전달

- CODE

```
int send_sock;
int bcast = 1; // SO_BROADCAST의 옵션정보를 1로 변경하기 위한 변수 초기화
send_sock = socket(PF_INET, SOCK_DGRAM, 0);
setsockopt( send_sock, SOL_SOCKET, SO_BROADCAST, (void*)&bcast, sizeof(bcast));
```

소켓과 표준 입출력

표준 입출력 함수의 장점

- 표준 입출력 함수의 두 가지 장점
  1. 표준 입출력 함수는 이식성(Portability)이 좋다 // 표준 함수 자체적 장점
  2. 표준 입출력 함수는 버퍼링을 통한 성능의 향상에 도움이 된다.