2) 소켓 주소

네트워크 계층 패킷에서 발신지 포트와 목적지 포트가 필요하다. 이 주소 정보를 소켓 라이브러리와 주고받기 위해 API에 **sockaddr 자료형이 정의**되어 있다.

struct sockaddr

{

unit16\_t sa\_family; // 주소의 종류를 나타내는 상수값 4/6 byte

char sa\_data[14]; // 데이터

};

**Ipv4 패킷용 주소**를 만들려면 sockaddr\_in형을 사용한다.

struct sockaddr\_in

{

short sin\_family; // 주소의 종류를 나타내느 상수값 4/6 byte

uint16\_t sin\_port; // 포트값 16bit

struct in\_addr in\_addr; // 접속할 주소

char sin\_zero[8];

// sin\_zero는 사용하지 않으며 sockaddr\_in 구조체의 크기를 sockaddr과 맞추기 위한 패딩값 0으로 모두 채워야 함

};

struct in\_addr

{

union

{

struct

{

uint8\_t s\_b1, s\_b2, s\_b3, s\_4;

} S\_un\_b;

struct

{

unit16\_t s\_w1, s\_w2;

} S\_un\_w;

uint32\_t S\_addr;

} S\_un;

};

uint16\_t htons(uint16\_t hostshort) : 부호 없는 16비트 정수를 받아 호스트의 네이티브 바이트 순서에서 네트워크 바이트 순서로 변환

uint32\_t htonl(uint32\_t hostlon) : 32비트 정수를 받아 호스트의 네이티브 바이트 순서로 변환

**IP 조소는 65.254.248.180 포트는 80번으로 지정 sockaddr\_in 구조체 초기화**

sockaddr\_in myAddr;

memset(myAddr.sin\_zero, 0, sizeof(myAddr.sin\_zero));

myAddr.sin\_faily = AF\_INET;

myAddr.sin\_port = htons(80);

myAddr.sin\_addr.S\_un.S\_un\_b.s\_b1 = 65;

myAddr.sin\_addr.S\_un.S\_un\_b.s\_b1 = 254;

myAddr.sin\_addr.S\_un.S\_un\_b.s\_b1 = 248;

myAddr.sin\_addr.S\_un.S\_un\_b.s\_b1 = 180;