TI DSP, MCU 및 Xilinx Zynq FPGA 프로그래밍 전문가 과정

강사 – Innova Lee(이상훈)
gcccompil3r@gmail.com
학생 – 최대성
c3d4s19@naver.com

2018.03.30. – 027일차 수업

---

* 인터넷 주소 관련 함수들 예제

inet_aton() 함수 예제

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>
typedef struct sockaddr_in si;
void err_handler(char* msg){
        write(2, msg, strlen(msg));
        exit(1);
}

int main(int argc, char** agrv){
        char* addr = "127.124.73.31";
        si addr_inet;
        if(!inet_aton(addr, &addr_inet.sin_addr))
                err_handler("Conversion Error!");
        else
                printf("Network Ordered Integer
Addr: %#xWn", addr_inet.sin_addr.s_addr);
        return 0;
}
```

inet_ntoa() 함수 예제

```
#include <stdio.h>
#include <string.h>
#include <arpa/inet.h>
typedef struct sockaddr_in si;

int main(int argc, char** argv){
        si addr1, addr2;
        char* str;
        char str_arr[32] = {0};

        addr1.sin_addr.s_addr = htonl(0x10203040);
        addr2.sin_addr.s_addr = htonl(0x12345678);

        str = inet_ntoa(addr1.sin_addr);
        strcpy(str_arr, str);
        printf("Not 1: %sWn", str);

        inet_ntoa(addr2.sin_addr);
        printf("Not 2: %sWn", str);
        printf("Not 3: %sWn", str_arr);
        return 0;
}
/*컴파일러가 딥러닝을 해서 str에 직접 주소값을
넣어주지 않아도 알아서 처리해준다 물론 이런식으로
코딩하면 안됨 그냥 컴파일러가 딥러닝 한다는 것을
보여주려는 예제*/
```

sockaddr_in 구조체의 주소멤버의 데이터 타입은 unsigned long이다

inet_addr() 함수

-> 10진수 표현방식(Dotted-Decimal Notation)의 주소값(문자열)을 비트 unsigned long타입의 Big Endian(네트워크 바이트 순서)로 변환한다.

unsigned long inet_addr(const char* [string])

-> 성공시 Big Endian 32비트, 실패시 -1 반환

inet_aton() 함수

-> 10진수 표현방식(Dotted-Decimal Notation)의 주소값(문자열)을 32비트 Big Endian(네트워크 바이트 순서)로 변환한다.

(inet_addr() 함수와 같은 기능, 두번째 인자 [addr]에 들어가는 sockaddr_in.in_addr구조체에 변환 값을 알아서 넣어준다)

int inet_aton(const char* [string], struct in_addr* [addr])

-> 성공시 1, 실패시 0 반환

inet_ntoa() 함수

-> int타입의 Big Endian(네트워크 바이트 순서)를 10진수 표현방식(Dotted-Decimal Notation)의 문자열로 변환

(inet_aton() 함수와 반대되는 기능, sockaddr_in.in_addr 구조체에 있는 값을 변환하여 문자열의 주소값을 리턴한다)

char* inet_ntoa(struct in_addr addr)

-> 변환된 해당 문자열의 포인터 반환, 실패시 -1

클라이언트에서 보낸 계산 요청을

서버에서 처리하여 보내주는 예제

<op.server.c 파일>

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <sys/socket.h>

typedef struct sockaddr_in si;
typedef struct sockaddr* sap;

#define BUF_SIZE    1024
#define OPSZ        4       //Operaton Size

void err_handler(char* msg){
    fputs(msg, stderr);
    fputc('Wn', stderr);
    exit(1);
}

int calculate(int opnum, int* opnds, char op){
    int result = opnds[0], i;

    switch(op){
        case '+':
            for(i = 1; i < opnum; i++)
                result += opnds[i];
            break;
        case '-':
            for(i = 1; i < opnum; i++)
                result -= opnds[i];
            break;
        case '*':
            for(i = 1; i < opnum; i++)
                result *= opnds[i];
            break;
    }
    return result;
}

int main(int argc, char** argv){
    int serv_sock, clnt_sock;
    char opinfo[BUF_SIZE];

    int result, opnd_cnt, i;
    int recv_cnt, recv_len;

    si serv_addr, clnt_addr;
    socklen_t clnt_addr_size;

    if(argc != 2){
        printf("use: %s <port>Wn",argv[0]);
        exit(1);
    }

    serv_sock = socket(PF_INET, SOCK_STREAM, 0);

    if(serv_sock == -1)
        err_handler("socket() error");

    memset(&serv_addr, 0, sizeof(serv_addr));
    serv_addr.sin_family = AF_INET;
    serv_addr.sin_addr.s_addr = htonl(INADDR_ANY);
    serv_addr.sin_port = htons(atoi(argv[1]));

    if(bind(serv_sock, (sap)&serv_addr,
sizeof(serv_addr)) == -1)
        err_handler("bind() error");

    if(listen(serv_sock, 5) == -1)
        err_handler("listen() error");

    clnt_addr_size = sizeof(clnt_addr);

    for(i = 0; i < 5; i++){ //여러명 접속 대비
        opnd_cnt = 0;
        clnt_sock = accept(serv_sock,
(sap)&clnt_addr, &clnt_addr_size);
        read(clnt_sock, &opnd_cnt, 1);

        recv_len = 0;

        while((opnd_cnt * OPSZ + 1) > recv_len){
            recv_cnt = read(clnt_sock,
&opinfo[recv_len], BUF_SIZE -1);
            recv_len += recv_cnt;
        }

        result = calculate(opnd_cnt, (int*)opinfo,
opinfo[recv_len - 1]);
        write(clnt_sock, (char*)&result,
sizeof(result));

        close(clnt_sock);
    }
    close(serv_sock);
}
```

<op.server.c 파일>

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <sys/socket.h>

typedef struct sockaddr_in si;
typedef struct sockaddr* sap;

#define BUF_SIZE    1024
#define OPSZ        4       //Operaton Siz
#define RLT_SIZE    4

void err_handler(char* msg){
    fputs(msg, stderr);
    fputc('₩n', stderr);
    exit(1);
}

int main(int argc, char** argv){
    int i , sock, result, opnd_cnt;
    char opmsg[BUF_SIZE] = {0};
    si serv_addr;

    if(argc != 3){
        printf("use: %s <IP> <port>₩n", argv[0]);
        exit(1);
    }

    sock = socket(PF_INET, SOCK_STREAM, 0);

    if(sock == -1)
        err_handler("socket() error");

    memset(&serv_addr, 0, sizeof(serv_addr));
    serv_addr.sin_family = AF_INET;
    serv_addr.sin_addr.s_addr =
inet_addr(argv[1]);
    serv_addr.sin_port = htons(atoi(argv[2]));

    if(connect(sock, (sap)&serv_addr,
sizeof(serv_addr)) == -1)
        err_handler("connect() error");
    else
        puts("Connected .............");

    fputs("Operand Cnt: ",stdout );
    scanf("%d", &opnd_cnt);

    opmsg[0] = (char)opnd_cnt;

    for(i = 0; i < opnd_cnt; i++){
        printf("Operand %d: ", i + 1);
        scanf("%d", (int*)&opmsg[i * OPSZ + 1]);
    }
    fgetc(stdin);
    fputs("Operator: ",stdout);
    scanf("%c", &opmsg[opnd_cnt * OPSZ + 1]);
    write(sock, opmsg, opnd_cnt * OPSZ + 2);
    read(sock, &result, RLT_SIZE);

    printf("Operation result: %d₩n", result);
    close(sock);

    return 0;
}
```

-> 통신 부분의 경우 패턴이 비슷하므로 패턴을 익혀두면 편하다.