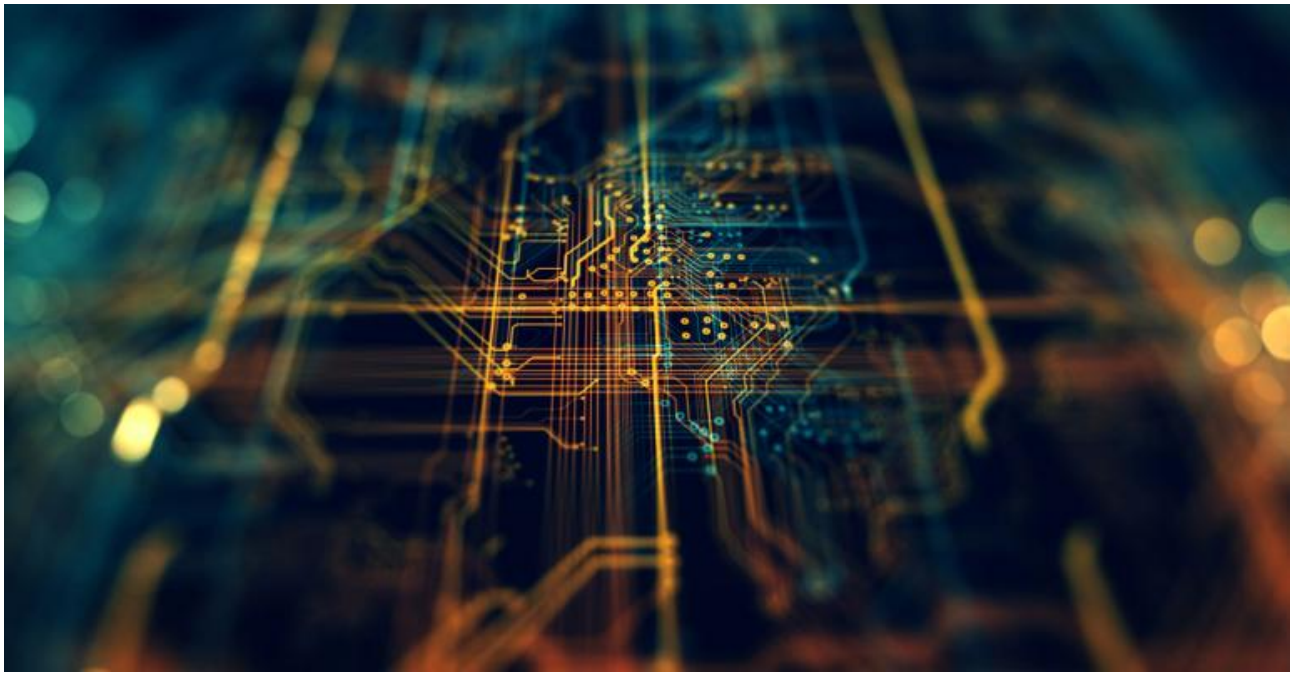
TI DSP, MCU 및 Xilinx Zynq FPGA 프로그래밍 전문가 과정



2018.03.30
27 일차
강사 – Innova Lee(이상훈)
gcccompil3r@gmail.com

학생 – 신민철
akrn33@naver.com

echo_client.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

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

int main(int argc, char** argv)
{
	int sock, str_len;
	si serv_addr;
	char msg[32];
	char* m = "input Message9q to quit): ";

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

	sock = socket(PF_INET, SOCK_STREAM, 0);
	//1 번 인자 int domain: 소켓이 사용할 프로토콜 체계를 선택
```

```
	//2 번 인자 int type : 소켓의 데이터 전송 방식에 대한 정보 전달
	//3 번 인자 int protocol : 두 컴퓨터간 통신에 사용되는 프로토콜 정보 전달.
	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() errer");
	else
		puts("Connected .....");
	for(;;)
	{
		fputs("Input msg(q to quit): ", stdout);
	fgets(msg, BUF_SIZE, stdin);
		if(!strcmp(msg, "q\n") || !strcmp(msg, "Q\n"))
		break;

	write(sock, msg, strlen(msg));
	str_len = read(sock, msg, BUF_SIZE -1);

	msg[str_len] = 0;

	printf("msg from serv: %s\n", msg);

	}
	close(sock);
	return 0;
}
```

echo_server.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

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


int main(int argc, char**argv)
{
	int i, str_len;
	int serv_sock, clnt_sock;

	char msg[BUF_SIZE];

	si serv_addr, clnt_addr;
	socklen_t clnt_addr_size;

	if(argc != 2)
	{
		printf("use: %s <port>\n", 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]));
    //bind 함수의 첫번째 인자에는 serv_sock 의 반환값,
    //두번째 인자에는 바인드할 소켓 주소가 표기된다.
    //마지막 매개변수는 두번째 인자인 주소의 크기다.
    if(bind(serv_sock,  (sap)&serv_addr,  sizeof(serv_addr))  ==
-1)
        err_handler("bind() error");
    //
    if(listen(serv_sock, 5) == -1)
        err_handler("listen(0 error");
    clnt_addr_size = sizeof(clnt_addr);

    for(i = 0; i < 5; i++)
    {
        clnt_sock          =          accept(serv_sock,(struct
sockaddr*)&clnt_addr, &clnt_addr_size);
        printf("clnt_sock = %d\n",clnt_sock);

        if(clnt_sock == -1)
            err_handler("accept() error");
        else
            printf("Connected Client %d\n", i + 1);

        while((str_len = read(clnt_sock, msg, BUF_SIZE)) != 0)
```

```
                write(clnt_sock, msg, str_len);

            close(clnt_sock);
        }

        close(serv_sock);
        return 0;
}
```

op_client.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 RLT_SIZE 4
#define OPSZ        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("Operration resutl: %d\n", result);
	close(sock);

	return 0;
	return 0;
}
```

op_server.c//네트워크를 통한 계산기

```
#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//오퍼레이션의 사이즈 4 개를 허용하겠다.

void err_handler(char* msg)
{
	fputs(msg, stderr);
	fputc('\n', 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>\n",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))
		err_handler("listen() error");

	clnt_addr_size = sizeof(clnt_addr);

	for(i = 0; i< 5; i++)
	{
		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);
		return 0;
	}
}
```

inet_nota.c//리틀엔디안을 빅엔디안으로 바꿔주는 코드

```
//networkaddress
#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: %s\n", str);

	inet_ntoa(addr2.sin_addr);
	printf("Not 2: %s\n",str);
	printf("Not 3: %s\n",str_arr);

	return 0;
}
```

inet_aton.c// 빅엔디안의 주소를 리틀엔디안 16 진수로 바꿔주는코드

```
#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** argv)
{
	char* addr = "127.124.88.9";
	si addr_inet;

	if(!inet_aton(addr, &addr_inet.sin_addr))
		err_handler("Conversion Error!");
	else
		printf("Network Ordered Integer Addr: %#x\n",
addr_inet.sin_addr.s_addr);
	return 0;
```

}