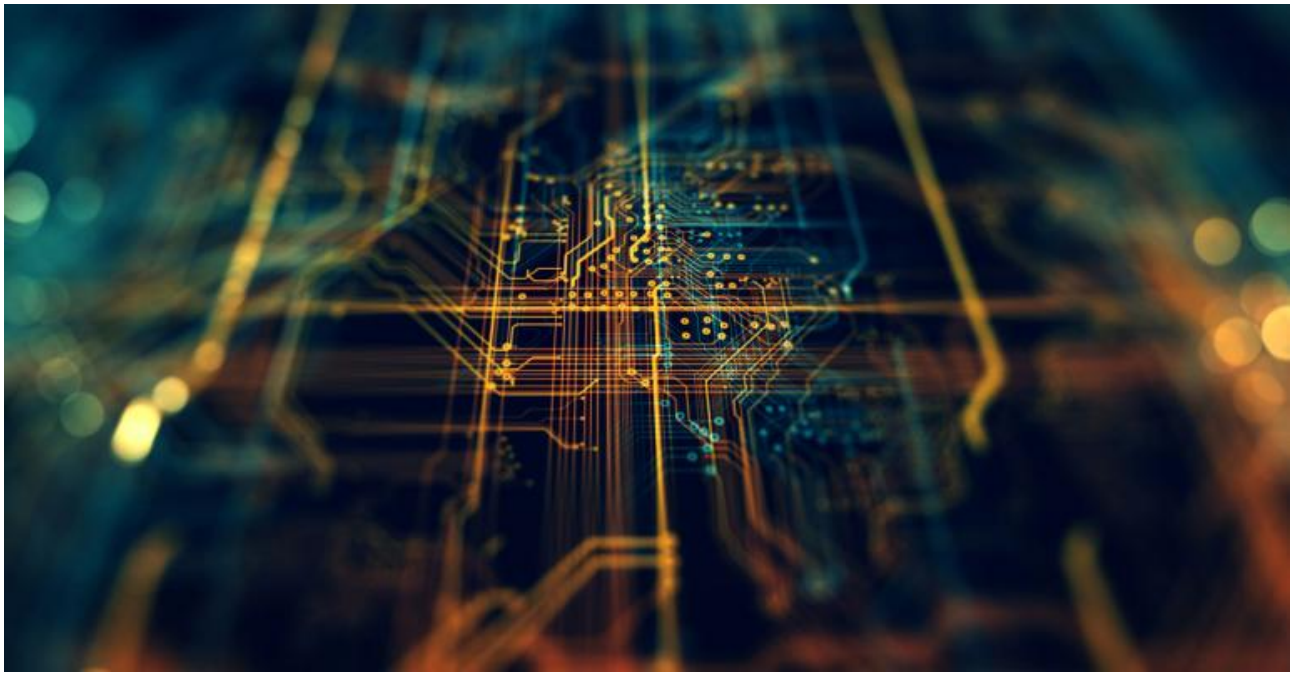
TI DSP, MCU 및 Xilinx Zynq FPGA 프로그래밍 전문가 과정



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

학생 – 신민철
akrn33@naver.com

gserv.c

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

#define    BUF_SIZE     128
#define MAX_CLNT     256

typedef struct sockaddr_in          si;
typedef struct sockaddr *           sp;

int clnt_cnt = 0;
int clnt_socks[MAX_CLNT];
int data[MAX_CLNT];
int thread_pid[MAX_CLNT];
int idx;
int cnt[MAX_CLNT];
pthread_mutex_t mtx;//락을 걸건데 락의 키값을 받

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

void sig_handler(int signo)
{
      int i;
```

```
	printf("Time Over!\n");

	pthread_mutex_lock(&mtx);

	for(i = 0; i < clnt_cnt; i++)
		if(thread_pid[i] == getpid())
			cnt[i] += 1;

	pthread_mutex_unlock(&mtx);

	alarm(3);
}

void proc_msg(char *msg, int len, int k)
{
	int i;
	int cmp = atoi(msg);
	char smsg[64] = {0};

	pthread_mutex_lock(&mtx);

	cnt[k] += 1;

	if(data[k] > cmp)
		sprintf(smsg, "greater than %d\n", cmp);
	else if(data[k] < cmp)
		sprintf(smsg, "less than %d\n", cmp);
	else
	{
		strcpy(smsg, "You win!\n");
		printf("cnt = %d\n", cnt[k]);
	}

	strcat(smsg, "Input Number: \n");
	write(clnt_socks[k], smsg, strlen(smsg));
```

```
#if 0
	for(i = 0; i < clnt_cnt; i++)
	{
		if(data[i] > cmp)
			sprintf(smsg, "greater than %d\n", cmp);
		else if(data[i] < cmp)
			sprintf(smsg, "less than %d\n", cmp);
		else
			strcpy(smsg, "You win!\n");

		strcat(smsg, "Input Number: ");
		write(clnt_socks[i], smsg, strlen(smsg));
	}
#endif

	pthread_mutex_unlock(&mtx);
}

void *clnt_handler(void *arg)//쓰레드의 마지막인자값이 들어옴
{
	int clnt_sock = *((int *)arg);
	int str_len = 0, i;
	char msg[BUF_SIZE] = {0};
	char pattern[BUF_SIZE] = "Input Number: \n";

	signal(SIGALRM, sig_handler);

	pthread_mutex_lock(&mtx);
	thread_pid[idx++] = getpid();//clnt 핸들러는 쓰레드 안에서 동작하니까 쓰레드의 자식프로세스아이디값을 받아서 넣음
	i = idx - 1;
	printf("i = %d\n", i);
```

```
	write(clnt_socks[i], pattern, strlen(pattern));//첫 번째 클라이언트에다가 패턴을 써주겠다.
	pthread_mutex_unlock(&mtx);//뮤텍스언락을 풀어준다.

	alarm(3);//3 초대기

	while((str_len = read(clnt_sock, msg, sizeof(msg))) != 0)
	{
		alarm(0);
		proc_msg(msg, str_len, i);
		alarm(3);
	}

	pthread_mutex_lock(&mtx);

	for(i = 0; i < clnt_cnt; i++)
	{
		if(clnt_sock == clnt_socks[i])
		{
			while(i++ < clnt_cnt - 1)
				clnt_socks[i] = clnt_socks[i + 1];
			break;
		}
	}

	clnt_cnt--;
	pthread_mutex_unlock(&mtx);
	close(clnt_sock);

	return NULL;
}

int main(int argc, char **argv)
{
```

```
int serv_sock, clnt_sock;
si serv_addr, clnt_addr;
socklen_t addr_size;
pthread_t t_id;
int idx = 0;

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

srand(time(NULL));

pthread_mutex_init(&mtx, NULL);//락건게 없으니까 초기화시켜주는거

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, (sp)&serv_addr, sizeof(serv_addr)) == -1)
	err_handler("bind() error");

if(listen(serv_sock, 2) == -1)
	err_handler("listen() error");

for(;;)
```

```
	{
		addr_size = sizeof(clnt_addr);
		clnt_sock = accept(serv_sock, (sp)&clnt_addr, &addr_size);//클라이언트의 커넥트를 승인

		thread_pid[idx++] = getpid();//

		pthread_mutex_lock(&mtx);//락걸려있으니까접근하지마
		data[clnt_cnt] = rand() % 3333 + 1;
		clnt_socks[clnt_cnt++] = clnt_sock;
		pthread_mutex_unlock(&mtx);//다했으니까 락풀거야 할거면 해

		pthread_create(&t_id, NULL, clnt_handler, (void *)&clnt_sock);//3 번째인자는 쓰레드 자체가되는거,4 번째 쓰레드에전달되는인자
		pthread_detach(t_id);//티아이디는 쓰레드의 식별자같은거
		printf("Connected Client IP: %s\n", inet_ntoa(clnt_addr.sin_addr));
	}

	close(serv_sock);

	return 0;
}
```

gclnt.c

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

#define BUF_SIZE        128

typedef struct sockaddr_in     si;
typedef struct sockaddr *      sp;

char msg[BUF_SIZE];

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

void *send_msg(void *arg)//피쓰레드크리에이트의 마지막인자 인자 10 개로 받으려면 구조체로 묶어서 보내야한다.
{
        int sock = *((int *)arg);
        char msg[BUF_SIZE];

        for(;;)
        {
                fgets(msg, BUF_SIZE, stdin);
```

```
            write(sock, msg, strlen(msg));
        }

        return NULL;
}

void *recv_msg(void *arg)
{
        int sock = *((int *)arg);
        char msg[BUF_SIZE];
        int str_len;

        for(;;)
        {
            str_len = read(sock, msg, BUF_SIZE - 1);

            msg[str_len] = 0;
            fputs(msg, stdout);
        }

        return NULL;
}

int main(int argc, char **argv)
{
        int sock;
        si serv_addr;
        pthread_t snd_thread, rcv_thread;
        void *thread_ret;

        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, (sp)&serv_addr, sizeof(serv_addr)) == -1)//액셉트를동작하게 해주는애 송신과 수신을 구분하기위해 쓰레드를 사용
		err_handler("connect() error");

	pthread_create(&snd_thread, NULL, send_msg, (void *)&sock);//메세지보내는 스레드 네번째 인자는 쓰레드가 구동시키는 함수의 인자
	pthread_create(&rcv_thread, NULL, recv_msg, (void *)&sock);//메세지 받는 스레드
	pthread_join(snd_thread, &thread_ret);//조인을 하는 순간 구동 더이상 센드할게 없거나 리시브할게 없거나 죽었을때 끝난다.
	pthread_join(rcv_thread, &thread_ret);

	close(sock);

	return 0;
}
```

gethostbyname.c//도메인네임서버의 아이피주소를 받는코드

```
#include<stdio.h>
#include<unistd.h>
#include<netdb.h>
#include<arpa/inet.h>
#include<stdlib.h>
void err_handler(char* msg)
{
	fputs(msg, stderr);
	fputc('\n',stderr);
	exit(1);
}

int main(int argc, char**argv){
	int i;
	struct hostent* host;

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

	host = gethostbyname(argv[1]);

	if(!host)
		err_handler("gethost ... error!");

	printf("Official Name: %s\n", host->h_name);

	for(i = 0 ; host->h_aliases[i]; i++)
		printf("Aliases %d: %s\n", i + 1, host->h_aliases[i]);

	printf("address Type: %s\n",
```

```
            (host->h_addrtype == AF_INET) ? "AF_INET" :
"AF_INET6");

        for(i = 0; host->h_addr_list[i]; i++)
                printf("IP Addr %d: %s\n", i + 1,
                        inet_ntoa(*(struct in_addr *)host->h_addr_list[i]));

        return 0;
}
```

file_client.c//파일 전송받는 클라이언트 코드

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

typedef struct sockaddr_in si;
typedef struct sockaddr * sap;

#define BUF_SIZE 32

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

int main(int argc, char** argv)
{
        char buf[BUF_SIZE] = {0};
        int fd, sock, read_cnt;
```

```
    si sock_addr,serv_addr;

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

    fd = open("receive.tt",O_CREAT|O_WRONLY);
    sock = socket(PF_INET, SOCK_STREAM, 0);

    if(sock == -1)
        err_handler("socket(0 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 .....");

    while((read_cnt = read(sock, buf, BUF_SIZE)) != 0)
        write(fd, buf, read_cnt);

    puts("Received File Data");
    write(sock, "Thank you",10);
    close(fd);
    close(sock);

    return 0;
}
```

file_server.c//파일을 전송해주는 서버 코드

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

typedef struct sockaddr_in si;
typedef struct sockaddr * sap;

#define BUF_SIZE 32

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

int main(int argc, char **argv)
{
	int serv_sock, clnt_sock, fd;
	char buf[BUF_SIZE] = {0};
	int read_cnt;

	si serv_addr, clnt_addr;
	socklen_t clnt_addr_size;

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

```
	fd = open("file_server.c", O_RDONLY);
	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);

	clnt_sock = accept(serv_sock, (sap)&clnt_addr, &
clnt_addr_size);
	printf("clnt_sock ======  %d =======  \n",clnt_sock);
	for(;;)
	{
		read_cnt = read(fd, buf, BUF_SIZE);

		if(read_cnt < BUF_SIZE)
		{
			write(clnt_sock, buf, read_cnt);//클라이언트소켓에다
가 씀(전송부분)
			break;
		}
```

```
            write(clnt_sock, buf, BUF_SIZE);
        }
        shutdown(clnt_sock, SHUT_WR);
        read(clnt_sock, buf, BUF_SIZE);
        printf("msg from client: %s\n", buf);

        close(fd);
        close(clnt_sock);
        close(serv_sock);

        return 0;
}
```