Xilinx Zynq FPGA, TI DSP MCU 기반의

프로그래밍 및 회로 설계 전문가

강사 이상훈

(Innova Lee)

Gcccompil3r@gmail.com

학생 김민호

minking12@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;

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);//들어온 msg 에 atoi ->숫자값으로 변환
	char smsg[64] = {0};

	pthread_mutex_lock(&mtx);

	cnt[k] += 1;//카운팅 값을 해주고 있다 이 뜻은 몇번 입력했는지

	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");//strcat 은 string 에 갖다 붙힌다.
	write(clnt_socks[k], smsg, strlen(smsg));//클라이언트 소켓에 전달해준다.

#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));//클라이언트 소켓에 정보를 전달
	}
#endif

	pthread_mutex_unlock(&mtx);//작업이 끝났으니까 unlock 해준다.
}

void *clnt_handler(void *arg)//arg 에 pthread_create 의 마지막인자가 들어옴 (clnt socket(fd))
{
	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);//3 초내에 등록하지 않으면 핸들러에 등록시킨다. sig_handler 는 thread 에서 동작 걔를 pid 하면 스레드의 pid 값이 나온다.
```

```
	pthread_mutex_lock(&mtx);//다시 락을 건다. 크리티컬 영역에 빠지지 않도록 락. . . . . .
	thread_pid[idx++] = getpid();//pid 에 getpid 에 넣는다.
	i = idx - 1;//현재 인덱스 값을 확인하기 위해 -1 한다
	printf("i = %d\n", i);
	write(clnt_socks[i], pattern, strlen(pattern));//clnt_sock 은 idx 는 0 부터 시작하는데 첫번째 클라이언트에다가 패턴을 써주겠다.
	pthread_mutex_unlock(&mtx);//다시 락을 푼다. 꼬이지 않도록

	alarm(3);//3 초 대기 . . . .

	while((str_len = read(clnt_sock, msg, sizeof(msg))) != 0)//write 로 패턴을 보냈다(숫자 입력하시오) read 는 클라이언트가 숫자를 입력했을때 무엇인지 수신을한다 -> msg 로 들어간다.
	{
		alarm(0);//알람 들어왔으니 초기화
		proc_msg(msg, str_len, i);//proc msg -> i 값들어오고 전달 proc msg 로
		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;//쓰레드 id 값
	int idx = 0;

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

```
	srand(time(NULL));

	pthread_mutex_init(&mtx, NULL);//전역변수가 있다. -> 초기에는 락을 한게 없으니까 null 로 초기화.

	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)//2 명 받음
		err_handler("listen() error");

	for(;;)
	{
		addr_size = sizeof(clnt_addr);
		clnt_sock = accept(serv_sock, (sp)&clnt_addr, &addr_size);//클라이언트의 접근을 승인-> 리턴결과는 클라이언트의 소켓(fd) -> 다음 클라이언트가 올때까지 블로킹 -> clnt_handler 로

		thread_pid[idx++] = getpid();//pid 값을 넣고있다.

		pthread_mutex_lock(&mtx);//락을 거는 이유는 데이타가 꼬이지 말라고.
		data[clnt_cnt] = rand() % 3333 + 1;
		clnt_socks[clnt_cnt++] = clnt_sock;
		pthread_mutex_unlock(&mtx);//락을 풀어서 작업.

		pthread_create(&t_id, NULL, clnt_handler, (void *)&clnt_sock);//피 스레드를 만들고 아이디 값을 주었고, handler 는 스레드가 구동시키는 함수이다.(그 자체)-> 4 번째는 쓰레드에 전달되는 인자
		pthread_detach(t_id);//생성한 피 쓰레드를 떼어낸다 ? t_id 는 피쓰레드의 id 값 ,프로세스와 분리 시킨다는 말 이다.(cpu 에 할당)
		printf("Connected Client IP: %s\n", inet_ntoa(clnt_addr.sin_addr));//한명이 접속 했으니 아이피 값을 뿌려줌. accept 로 간다
	}

	close(serv_sock);

	return 0;
}
```

Gclient.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)//이 부분에 서버가 연결된 소켓이 인자로 넘어온다. (pthread_create 의 마지막인자는 구동시키려하는 함수의 인자값) -> 구조체로 만들면 좋다. 너무 많아서..
{
        int sock = *((int *)arg);//만들어진 서버의 소켓이 전달
        char msg[BUF_SIZE];

        for(;;)//무한루프를 돌면서 사용자의 입력을 받아오고 서버소켓에 전송해주는 스레드
        {
                fgets(msg, BUF_SIZE, stdin);//stdin : 입력을 받겠다. 그걸 msg 에 저장
                write(sock, msg, strlen(msg));//sock 에 write -> 우리가 입력한 메세지 값이 서버로 전송...
        }

        return NULL;
}

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

        for(;;)
        {
                str_len = read(sock, msg, BUF_SIZE - 1);//서버로부터 들어온 정보를 수신해서 msg 에 넣어준다.
```

```
            msg[str_len] = 0;
            fputs(msg, stdout);//넣어준 msg 를 모니터에 출력=write 0 과 같다.
        }

        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)//소켓 하고 connect 하는 순간
서버의 accept 가 동작한다 .
            err_handler("connect() error");

        pthread_create(&snd_thread, NULL, send_msg, (void *)&sock);//쓰레드를 만든 이유는 송
신과 수신을 분리하기 위하여 이다. 이는 fork 와 같다. (snd ~ 메세지 보내는 쓰레드)(send msg 함수에서
처리 인자로 둘다 sock 사용)
        pthread_create(&rcv_thread, NULL, recv_msg, (void *)&sock);//수신하는 쓰레드
        pthread_join(snd_thread, &thread_ret);//join 하는 순간 쓰레드는 동작을 시작한다. join 은
더이상 송,수신이나 쓰레드가 없으면 끝난다. ctrl +c 를 눌렀을때도.
        pthread_join(rcv_thread, &thread_ret);

        close(sock);

        return 0;
}
```

file_server.c

```
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <signal.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);//file_server.c 를 열고있다. 뭔가 전송하고 싶으면 여기를 바꿔야
한다.(fd 값을 받아옴)
   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);

    for(;;)
    {
        read_cnt=read(fd,buf,BUF_SIZE);//fd 에서 읽고 있으니 아까 오픈한 file_sever.c 에서 읽고 있다.
buf 에 넣는다.

        if(read_cnt<BUF_SIZE)//마무리가 덜 되었으면 -> read_cnt 는 읽은 바이트 수
        {
           write(clnt_sock,buf,read_cnt);//buf 를 clnt_sock 에 쓰고 있다.
           break;//클라이언트에게 보내는 정보는 file_server.c 라는 소리이다.
        }

        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;
}
```

file_client.c

```
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <signal.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)
{
   char buf[BUF_SIZE]={0};
   int fd,sock,read_cnt;
   si serv_addr;

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

   fd=open("receive.txt",O_CREAT|O_WRONLY,0644);//fd 로 연것이 file_server.c 이다.
   sock=socket(PF_INET,SOCK_STREAM,0);//클라이언트에 보낸다.

   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.....");
```

```
    while((read_cnt=read(sock,buf,BUF_SIZE))!=0)//서버로부터 받아온걸 읽오있다 이걸 fd 에 쓰고있다
receive.txt 에 file_server.c 내용이 들어가있으면 된다. 다른걸 전송하고 싶으면 서버에서 바꾸면 된다.
            write(fd,buf,read_cnt);

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

    return 0;
}
```

gethostbyname

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <pthread.h>
#include <arpa/inet.h>
#include <sys/socket.h>
#include <sys/epoll.h>
#include<unistd.h>
#include<arpa/inet.h>
#include<netdb.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]);//gethostbyname 1 번을 받고있다. 호스트로 리턴.

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

   printf("official name:%s\n",host->h_name);//호스트 이름

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

   printf("address type : %s\n",//ip 버전 4 인지 6 인지
          (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,//ip 주소값이 나온다. (웹의 포트번호 80 번으로 접속함.)
                inet_ntoa(*(struct in_addr *)host->h_addr_list[i]));

   return 0;
}
```