TI DSP,Xilinx zynq FPGA,MCU 및 Xilinx zynq FPGA 프로그래밍 전문가 과정

강사-INNOVA LEE(이상훈)
Gccompil3r@gmail.com
학생-윤지완
Yoonjw7894@naver.com

```
#include <stdio.h>
#include<arpa/inet.h>
#include<stdlib.h>
#include<unistd.h>
#include<string.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= "127.124.73.31";
si addr_inet;
if(!inet_aton(addr,&addr_inet.sin_addr))
err_handler("conversio)error!");
else
printf("network ordered intrger addr:%#x\n", addr_inet.sin_addr.s_addr);
return 0;
}
```

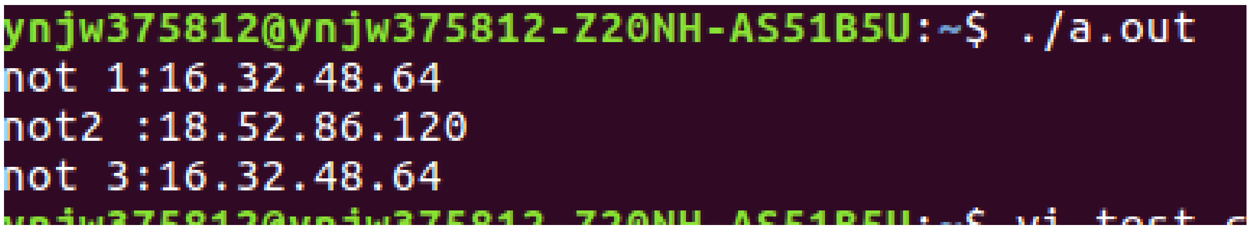
네트워크는 무조건 blg 이다.

```
호스트      네트워크     호스트
   리틀   →      빅    →   빅
    리틀   →    빅      → 리틀
```

```
#include<unistd.h>
#include<string.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);//4byte
addr2.sin_addr.s_addr = htonl(0x12345678);//16*1+2=18,16*3+4=52 이런식으로 계산한다,
이유는 두숫자가 1byte 이기 때문이다.
str = inet_ntoa(addr1.sin_addr);//위에 변환된 addr1 을 다시 네트워크에서 호스트로
 역변환하는 과정.
strcpy(str_arr ,str);
printf("not 1:%s\n", str);
inet_ntoa(addr2.sin_addr);
printf("not2 :%s\n",str);
printf("not3 :%s\n", str_arr);
return 0;
}
```



```
ynjw375812@ynjw375812-Z20NH-AS51B5U:~$ ./a.out
not 1:16.32.48.64
not2 :18.52.86.120
not 3:16.32.48.64
```

```
#include <stdio.h>
#include<arpa/inet.h>
#include<stdlib.h>
#include<unistd.h>
#include<string.h>
#include<sys/socket.h>
typedef struct sockaddr_in   si;
typedef struct sockaddr * sad;

#define BUF_SIZE    1024
void err_handler(char *msg)
{
fputs(msg, stderr);
fputs("\n", stderr);
exit(1);
}



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


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("accep() error");
mamset(&serv_addr,0,sizeof(serv_addr));
serv_addr.sin_family= AF_INET;
serv_addr.sin_addr.s_addr= hton(INADDR_ANY);
serv_addr.sin_port = hton(atoi(argv[1]));
```

```
if(bind(serv_sock , (sad)&serv_addr , sizeof(serv_addr)) == -1)
err_handler("bind() error");
if(listen(serv_sock , 5) == -1)
err_handler("listne() 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);
if(clnt_sock ==-1)
err_handler("accept() error");
else
printf("conneted clien %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;
}
fputs==write     fgets =read

<SERV>

#include <stdio.h>
#include<arpa/inet.h>
#include<stdlib.h>
#include<unistd.h>
#include<string.h>
#include<sys/socket.h>
typedef struct sockaddr_in   si;
typedef struct sockaddr * sad;

#define BUF_SIZE    1024
#define OPSZ    4//OPRATION 의 사이즈
void err_handler(char *msg)
{
fputs(msg, stderr);
fputc("\n", stderr);
exit(1);
}
int calculate(int onum, int *opnds ,char op)
{
int result = opnds[0],i;
switch(op)
{
case '+':
for(i=1;i<onum;i++)
result += opnds[i];
break;
case '-' :
```

```
for(i=1;i<onum;i++)
result += opnds[i];
break;
case '*':
for(i=1;i<onum;i++)
result += opnds[i];
break;
}
return result;
}


int main(int argc,char **argv)
{
int i,str_len;
int serv_sock,clnt_sock;
char opinfo[BUF_SIZE];
int result ,opnds,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[2]));

if(bind(serv_sock , (sad)&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++)
{
opnds_cnt = ;
clnt_sock = accept(serv_sock,(sad)&clnt_addr, &clnt_addr_size);
read(clnt_sock,&opnds_cnt,1);
recv_len= 0;
while((opnds_cnt * OPSZ +1)>ercv_len)
{
{
opnds_cnt = ;
clnt_sock = accept(serv_sock,(sad)&clnt_addr, &clnt_addr_size);
read(clnt_sock,&opnds_cnt,1);
recv_len= 0;
while((opnds_cnt * OPSZ +1)>ercv_len)
{
recv_cnt = read(clnt_sock,&opnds_cnt +1);
recv_len += recv_cnt;
}
result = calculate(opnds_cnt,(int *)opifo,opnifo[recv_len -1]);
close(clnt_sock);
}
clse(serv_sock);
return 0;
}

clnt_sock = accept(serv_sock,(struct  sockaddr *)&clnt_addr,&clnt_addr_size);
if(clnt_sock ==-1)
err_handler("accept() error");
else
printf("conneted clien %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;
}
```

<client>

```
#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 * sad;
#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 sock,i,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, (sad)&serv_addr, sizeof(serv_addr)) == -1)
      err_handler("connect() error");
    else
    puts("conneted .........");
fputs("operad cut: ",stdout);
scanf("%d",&opnd_cnt);//인자를 저장
opmsg[0]=(char )opnd_cnt;

for(i=0;i<opnd_cnt;i++)
{
printf("operad %d : ", stdout);
scanf("%d",(int *)&opmsg[i*OPSZ+1]);
}
fgetc(stdin);
fputs("oprator :",stdout);
scanf("%c",&opmsg[opnd_cnt * OPSZ +1]);
write(sock,opmsg,opnd_cnt * OPSZ+2);//인자를 받은거 부호모두가 opmsg 에 저
장된다.
read(sock,&result,RLT_SIZE);//서버에서 계산된 값을 보내서 클라이언트가 받
는다.
printf("operation result :%d\n",result);
close(sock);
return 0;
```

```
}
```

<serv>

```
#include <stdio.h>
#include<arpa/inet.h>
#include<stdlib.h>
#include<unistd.h>
#include<string.h>
#include<sys/socket.h>pid=fork();
#include<setjmp.h>
typedef struct sockaddr_in   si;
typedef struct sockaddr * sad;
jmp buf_env;
#define BUF_SIZE    1024
#define OPSZ 4
void err_handler(char *msg)
{
fputs(msg, stderr);//출력
fputc("\n", stderr);
exit(1);
}

int calculate(int onum, int *opnds ,char op)//인자의 수가 onum 에 저장
{
int result = opnds[0],i;
switch(op)

{
case '+':
for(i=1;i<onum;i++)
result += opnds[i];
break;
case '-' :
for(i=1;i<onum;i++)
result += opnds[i];
break;
case '*':
for(i=1;i<onum;i++)
result += opnds[i];
break;

}
return result;
}
void  count(int opnd_cnt)
{
if(opnd_cnt<ret){
```

```
printf(“up”\n);
}
else if(“opnd_cnt>ret)
{
printf(“DOWN\n”)
}
void choice(cp  *root)
{
root→num=rand()%10+1;
return root;
}


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;
choice(num);
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[2]));

if(bind(serv_sock , (sad)&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);

pid=fork();
if(pid==0)
{
printf(“%i\n”,getpid());
}
```

```
if(pid>0)
{

for(i=0;i<5;i++)
{
opnd_cnt =0 ;
clnt_sock = accept(serv_sock,(sad)&clnt_addr, &clnt_addr_size);
read(clnt_sock,&opnd_cnt,1);//1 바이트씩 클라이언트가 보낸 값을 읽는다.
count(opnds_cut);
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;//read 에서 읽은 byte 를 가지고 온다.그것을 더해서 저장한다.
//이 부분은 저장하다가 끊어지는 것을 방지.
}
result = calculate(opnd_cnt,(int *)opinfo,opinfo[recv_len -1]);
//배열은 처음이 0 부터니까 마지막은 -1 을 해주어야 한다.
write(clnt_sock,(char *)&result,sizeof(result));
close(clnt_sock);
}
close(serv_sock);
return 0;



for(;;)
	{
		nread = read(sock, buf, BUF_SIZE);
		write(1, buf, nread);

	}
```

이 부분에서는 setjmp 를 써서