TI DSP,MCU 및 Xilinux Zynq FPGA

프로그래밍 전문가 과정

이름	문지희
학생 이메일	mjh8127@naver.com
날짜	2018/3/22
수업일수	21 일차
담당강사	Innova Lee(이상훈)
강사 이메일	gcccompil3r@gmail.com

목차

1. 함수정리

-Vi 명령어 : touch
xeno@xeno-NH:~/proj/0322$ touch a.txt b.txt c.txt d.txt
xeno@xeno-NH:~/proj/0322$ ls
00.txt 0_1.c 1.c a.out a.txt b.txt c.txt d.txt

touch 옆에 쓴 a.txt b.txt c.txt d.txt 가 생성

-opendir( )
( )안의 디렉토리를 열고 그 디렉토리에 대한 포인터를 반환한다.

-readdir( )
개방한 디렉토리를 읽는다.

-dirent ( )
( )내에 있는 디렉토리 안의 리스트를 보여준다.

-stat( )
입력한 정보의 상태를 본다.
Ex) stat(argv[1],&buf) //정보를 buf 에 저장, argv 는 파일명이나 디렉토리명이 와야한다.

-getpwuid

id 를 얻는다.

ruid : real id

euid : effective id

-getcwd()

경로를 나타내는 문자열의 시작 주소가 반환된다.

Ex) getcwd(char *buf, size_t bufsize) //buf 는 현재 디렉토리의 경로가 저장될 배열, bufsize 는 현재 디렉토리 경로 문자열의 크기

2.파일 디렉토리

예제 1) ls 구현

```
#include<sys/types.h>
#include<dirent.h>
#include<stdio.h>

int main(void)
{
	DIR *dp;
	//파일포인터랑 비슷한 것
	int i=0;
	struct dirent *p;
	//dirent : 디렉토리 내에 있는 리스트
	dp = opendir(".");
	//현재 디렉토리 나타내는것 : ' . '

	while(p=readdir(dp)) //디렉토리에있는 리스트가 p 에 넘어옴
	{
		if(p->d_name[0] == '.')
			continue; //파일 이름의 첫번째가 ' . '이면 넘겨라
		printf("%-16s", p->d_name);
		if((i+1)%5==0) //5번 넘으면 개행
			printf("₩n");
		i++;
	}
	printf("₩n");
	closedir(dp);
	return 0;
}
```

~결과

```
xeno@xeno-NH:~/proj/0322$ ./a.out
a.out          2.c          c.txt          3.c          1.c
b.txt          a.txt        0_1.c          d.txt        00.txt
```

예제 2) ls 옵션 추가

```
#include<stdio.h>
#include<unistd.h>
#include<fcntl.h>

int main(int argc, char **argv)
{
        int cmd;
        while((cmd=getopt(argc,argv,"ab"))>0) //프로그램 실행 시 받은 인자를 옵션으로 사용.
                                                // a,b 를 옵션으로 사용 가능하다.
        {
                switch(cmd)
                {
                        case 'a':   // a 옵션 들어올 때
                                printf("a option₩n");
                                break;
                        case 'b':   // b 옵션 들어올 때
                                printf("b option₩n");
                                break;
                            default:            // a,b 가 아닌 다른 옵션이 들어올 때
                                printf("unkown option₩n");
                }
        }
```

```
        return 0;
}
```

~결과

```
xeno@xeno-NH:~/proj/0322$ ./a.out -a
a option
xeno@xeno-NH:~/proj/0322$ ./a.out -ba
b option
a option
xeno@xeno-NH:~/proj/0322$ ./a.out -abc
a option
b option
./a.out: invalid option -- 'c'
unkown option
xeno@xeno-NH:~/proj/0322$ ./a.out -b
b option
```

예제 3)

```
#include<dirent.h>
#include<sys/types.h>
#include<stdio.h>
#include<unistd.h>
#include<fcntl.h>

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

```
{
        DIR *dp;
        int i=0,cmd;
        struct dirent *p;
        cmd = getopt(argc,argv,"a");        //인자로 받은 것을 옵션으로 받는데 a 만 받는다.
        dp=opendir(".");  //현재 파일을 열음

        while(p=readdir(dp)) //dp 의 디렉토리 리스트를 읽음
        {
                if(cmd !='a') //받은 인자가 a 가 아닌 경우
                {
                        if(p->d_name[0]=='.')
                                continue;  // 숨김 파일인 경우 넘긴다.
                }
                printf("%-16s",p->d_name); //디렉토리 내의 파일 이름을 출력
                if((i+1)%5==0)
                        printf("₩n"); //5개일 때 마다 개행
                i++;
        }
        printf("₩n");
        closedir(dp);
        return 0;
}
```

~결과

```
xeno@xeno-NH:~/proj/0322$ ./a.out
a.out            2.c              c.txt            3.c              1.c
b.txt            a.txt            0_1.c            d.txt            00.txt
```

예제 4) 선생님 코드 해석

```
#include <stdio.h>
#include <unistd.h>
#define A          (1 << 0)
#define B          (1 << 1)
#define C          (1 << 2)
#define D          (1 << 3)
#define E          (1 << 4)
#define F          (1 << 5)
#define G          (1 << 6)
void check_flag(int flag)
{
        int i, tmp = flag;

        for(i = 0; i < 7; i++)
        {
                switch(tmp & (1 << i))
         //flag 에 들어온 인자 값에 따라 비트 수가 달라진다.
         //case 와 flag 가 같은 비트 수 에서 case 밑의 명령이 실행.

                {
                        case 1:
                                printf("A₩n");
                                break;
                        case 2:
                                printf("B₩n");
                                break;
                        case 4:
                                printf("C₩n");
                                break;
                        case 8:
                                printf("D₩n");
                                break;
                        case 16:
                                printf("E₩n");
                                break;
                        case 32:
                                printf("F₩n");
                                break;
                        case 64:
                                printf("G₩n");
                                break;
                }
        }
}
int main(int argc, char **argv)
{
        int cmd;
        int flag;

        /* 7 개의 옵션: a, b, c, d, e, f, g */
        while((cmd = getopt(argc, argv, "abcdefg")) > 0)
// 인자로 받은 것 중 a, b, c, d, e, f, g 만 옵션으로만 받을 것이다.
        {
                switch(cmd)
                {
                        case 'a':
                                flag |= 1 << 0;   // 1
//쉬프트 연산을 하고 flag 와 OR 연산을 한 뒤 flag 에 값을 넣는다.
```

```
                printf("a option₩n");
                break;
        case 'b':
                flag |= 1 << 1; // 10
                printf("b option₩n");
                break;
        case 'c':
                flag |= 1 << 2; //100
                printf("c option₩n");
                break;
        case 'd':
                flag |= 1 << 3; //1000
                printf("d option₩n");
                break;
        case 'e':
                flag |= 1 << 4; //10000
                printf("e option₩n");
                break;
        case 'f':
                flag |= 1 << 5; //100000
                printf("f option₩n");
                break;
        case 'g':
                flag |= 1 << 6; //100000
                printf("g option₩n");
                break;
        default:
                printf("unknown option₩n");
    }
}
```

```
        check_flag(flag);
        return 0;
}
```

옵션을 분할해서 사용가능

예제 5)

```
#include<stdio.h>
#include<dirent.h>
#include<fcntl.h>

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

	DIR *dp;
	int i=0,cmd;
	struct dirent *p;
	int flag=0;

	while((cmd=getopt(argc,argv,"alRi"))>0)
	{
		switch(cmd)
		{
			case 'a':
				flag |= 1;
				break;
			case 'l':
				flag |= 2;
				break;
			case 'R':
				flag |= 4;
				break;
			case 'i':
				flag |= 8;
				break;
		}
	}
	dp = opendir(".");
```

```
        while(p=readdir(dp))
        {
                if(!(flag & 1))       //a 옵션이있으면 거짓이되어 .을 제끼지 않음
                {
                        if(p->d_name[0]=='.')
                                continue;
                }
                printf("%-16s",p->d_name);
                if((i+1)%5==0)
                        printf("₩n");
        }
        printf("₩n");
        close(dp);
        return 0;
}
```

~결과

```
xeno@xeno-NH:~/proj/0322$ ./a.out
a.out           2.c             c.txt           3.c             1.c             b.txt           2-2.c           a.txt
0_1.c           2-3.c           d.txt           00.txt
xeno@xeno-NH:~/proj/0322$ ./a.out -a
a.out           2.c             .0_1.c.swp      c.txt           3.c             1.c             b.txt           .
2-2.c           a.txt           0_1.c           ..              2-3.c           d.txt           00.txt
```

예제 6)

```
#include<sys/types.h>
#include<sys/stat.h>
#include<unistd.h>
#include<stdio.h>

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

        struct stat buf;
        char ch;
        stat(argv[1],&buf);//stat : 상태를 본다.입력한 정보의 상태를 봄
                        //그 정보를 buf 에 저장
                        //argv 는 파일명이나 디렉토리명이 와야한다.
        if(S_ISDIR(buf.st_mode))//디렉토리
                ch='d';
        if(S_ISREG(buf.st_mode))//레귤러(일반) : 파일
                ch='-';
        if(S_ISFIFO(buf.st_mode))//파이프
                ch='P';
        if(S_ISLNK(buf.st_mode))//바로가기 파일
                ch='l';
        if(S_ISSOCK(buf.st_mode))//소켓(네트워크 관련 장치 주로 c 나 b 로 통합되어있다.)
                ch='s';
        if(S_ISCHR(buf.st_mode))//캐릭터 디바이스
                ch='c';
        if(S_ISBLK(buf.st_mode))//블록 디바이스
                ch='b';
        printf("%c₩n",ch);//입력된 정보의 상태를 출력

        return 0;
}
```

~결과
xeno@xeno-NH:~/proj/0322$ ./a.out 1.c
-

예제 7)

```
#include<sys/types.h>
#include<sys/stat.h>
#include<unistd.h>
#include<stdio.h>

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

        struct stat buf;

        char ch;
        char perm[11]="----------";
        char rwx[4]="rwx"; // read, write , execute
        int i;
        stat(argv[1],&buf);

        if(S_ISDIR(buf.st_mode))//디렉토리
                perm[0]='d';
        if(S_ISREG(buf.st_mode))//레귤러(일반) : 파일
                perm[0]='-';
        if(S_ISFIFO(buf.st_mode))//파이프
                perm[0]='P';
        if(S_ISLNK(buf.st_mode))//바로가기 파일
```

```
                perm[0]='l';
        if(S_ISSOCK(buf.st_mode))//소켓(네트워크 관련 장치 주로 c 나 b 로 통합되어있다.)
                perm[0]='s';
        if(S_ISCHR(buf.st_mode))//캐릭터 디바이스
                perm[0]='c';
        if(S_ISBLK(buf.st_mode))//블록 디바이스
                perm[0]='b';

        for(i=0;i<9;i++)//9자리 채워넣을 것
        {
                if((buf.st_mode>>(8-i))&1) //8번째부터 0번째 자리까지 0인지 1인지 본다.
                        perm[i+1]=rwx[i%3]; // 1이면 i 가 3의 배수가 아닐 때 perm [i+1]에 권한을 넣음
        }
        printf("%s₩n",perm); //perm[0]인 파일 형태와 root 의 사용자권한, 그룹권한, 제3자에 대한 권한을 표시한다.

        return 0;

}
```

~결과

```
xeno@xeno-NH:~/proj/0322$ ./a.out 00.txt
-rw-r--r--
```

rw → root 사용자권한
r-- → 그룹권한
r-- → 제3자에 대한 권한
사용자는 읽고쓰기 가능하고 그룹과 제3자는 읽기만 가능
파일종류 가 처음으로 온다.
파일 권한설정 하는데에 8진수사용. 8진수 사용할 땐 맨 첫번째에 0을 넣음

예제 8)

```
#include<sys/types.h>
#include<sys/stat.h>
#include<unistd.h>
#include<stdio.h>
#include<pwd.h>
#include<grp.h>

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

        struct stat buf;
        struct passwd *pw;
        struct group *gr;
        char ch;
        char perm[11]="----------";
        char rwx[4]="rwx";
        int i;
        stat(argv[1],&buf);

        if(S_ISDIR(buf.st_mode))//디렉토리
                perm[0]='d';
        if(S_ISREG(buf.st_mode))//레귤러(일반) : 파일
                perm[0]='-';
        if(S_ISFIFO(buf.st_mode))//파이프
                perm[0]='P';
        if(S_ISLNK(buf.st_mode))//바로가기 파일
                perm[0]='l';
        if(S_ISSOCK(buf.st_mode))//소켓(네트워크 관련 장치 주로 c 나 b 로 통합되어있다.)
                perm[0]='s';
        if(S_ISCHR(buf.st_mode))//캐릭터 디바이스
                perm[0]='c';
```

```
        if(S_ISBLK(buf.st_mode))//블록 디바이스
                perm[0]='b';

        for(i=0;i<9;i++)//9자리 채워넣을 것
        {
                if((buf.st_mode>>(8-i))&1)
                        perm[i+1]=rwx[i%3];
        }
        printf("%s₩n",perm);
        printf("%d",buf.st_nlink);
        pw=getpwuid(buf.st_uid); //real password id 를 pw 에 받음
        printf("%s",pw->pw_name);
        gr = getgrgid(buf.st_gid); // real group id 를 gr 에 받음
        printf("%s",gr->gr_name);


        return 0;

}
```

~결과

-rw-r—r--1xenoxeno
권한 / (숫자)nlink / 유저명 그룹명

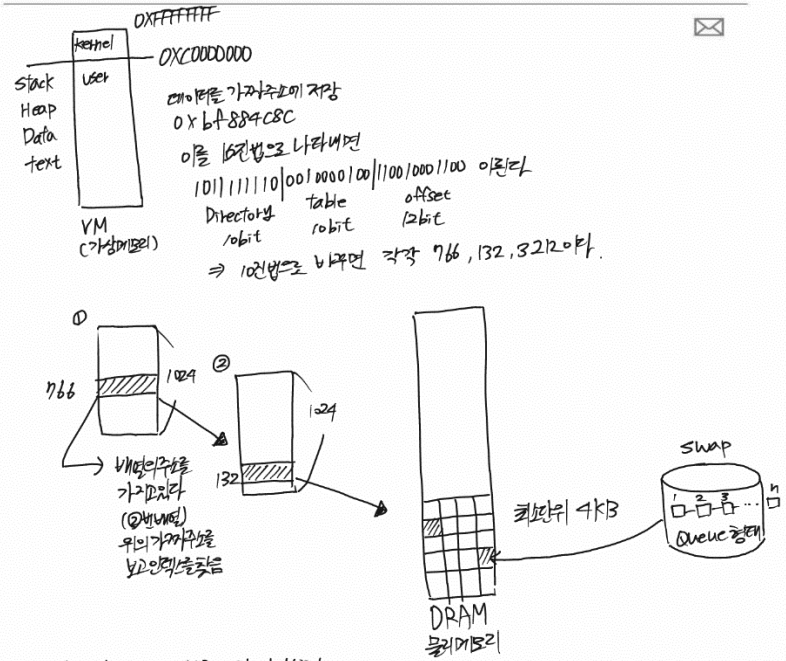
예제 9)

```
#include<sys/types.h>
#include<sys/stat.h>
#include<unistd.h>
#include<stdio.h>
#include<pwd.h>
#include<grp.h>
#include<time.h>

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

        struct stat buf;
        struct dirent *p;
        struct passwd *pw;
        struct group *gr;
        struct tm *tm;
        char ch;
        char perm[11]="----------";
        char rwx[4]="rwx";
        char sst[4]="sst";
        int i;
        stat(argv[1],&buf);

        if(S_ISDIR(buf.st_mode))//디렉토리
                perm[0]='d';
        if(S_ISREG(buf.st_mode))//레귤러(일반) : 파일
                perm[0]='-';
        if(S_ISFIFO(buf.st_mode))//파이프
                perm[0]='P';
        if(S_ISLNK(buf.st_mode))//바로가기 파일
                perm[0]='l';
```

```
	if(S_ISSOCK(buf.st_mode))//소켓(네트워크 관련 장치 주로 c 나 b 로 통합되어있다.)
		perm[0]='s';
	if(S_ISCHR(buf.st_mode))//캐릭터 디바이스
		perm[0]='c';
	if(S_ISBLK(buf.st_mode))//블록 디바이스
		perm[0]='b';

	for(i=0;i<9;i++)//9자리 채워넣을 것
	{
		if((buf.st_mode>>(8-i))&1)
			perm[i+1]=rwx[i%3];
	}
	for (i=0;i<3;i++)
	{
	if((buf.st_mode>>(11-i))&1) //앞에 3자리가 더 추가
		if(perm[(i+1)*3]=='.')
			perm[(i+1)*3]==sst[i]^0x20;
		else
			perm[(i+1)*3]=sst[i];
	}
	printf("%s₩n",perm);
	printf("%lu",buf.st_nlink);
	pw=getpwuid(buf.st_uid);
	printf("%s",pw->pw_name); //real password 이름 표시
	gr = getgrgid(buf.st_gid);
	printf("%s",gr->gr_name); //그룹이름 표시
	printf("%d",buf.st_size); // 사이즈 표시
	tm=localtime(&buff.st_mtime);
	printf("%d-%02d-%02d %02d %02d",tm->tm_year + 1900, tm->tm_mon+1,tm->tm_mday, tm->tm_hour, tm->tm_min); // 시간표시
	printf("₩n");
	return 0;
}
```

3. 가상 메모리구조

가상메모리에 값 저장하는 법



위와같은 방법으로 데이터를 저장하다보면
물리메모리에 더이상 저장할 공간이 부족해지게된다. 이때 LRU 알고리즘을사용

*LRU 알고리즘
물리메모리가 부족할때 디스크에위치한 swap의 힘을 빌린다
swap은 사용했던 메모리를 재활용한다. 자주쓰고, 없으면 안되는 것들을 넣는다. 필요할때마다 올리고 내린다.
메모리 계층구조에서는 메모리속도가 느린편이므로 현재필요한 정보만을 메이징하는것을 Demand On Paging 이라 한다.