Xilinx Zynq FPGA, TI DSP MCU 기반의

프로그래밍 및 회로 설계 전문가

강사 이상훈

(Innova Lee)

Gcccompil3r@gmail.com

학생 김민호

minking12@naver.com

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

/*

int main()     // ls 기능 구현.
{
    DIR *dp;//파일 포인터와 비슷, 파일 디렉토리
    int i=0;
    struct dirent *p;//디렉토리 내에 있는 리스트// 디렉토리 내에 있는 파일, 디렉토리 따위의
내용
    dp=opendir(".");//디렉토리도 오픈 해야 볼수있음 현재가 . 정보를 얻었고 // 반환값 : 오픈한
디렉토리의 주소값. "위치" 위치를 오픈
    while(p=readdir(dp))//디렉토리 내에 있는 리스트를 읽음 // READ는 DP의 안에 있는 내용을
첫번째 포인터, 그 다음 두번째. 그 다음 세번째, 쭉쭉 그다음 NULL
    {
      if(p->d_name[0]=='.')//배열의 0번은 맨앞 글자 0이면 재낀다.      .은  숨김파일// 리
눅스에서는 파일이름 앞에 .이 있을 경우 숨김파일로 간주, 즉 숨긴파일은 보지 않겠다는 의미
          continue;
      printf("%-16s ",p->d_name);//맨앞이 .이면 출력
      if((i+1)%5==0)//5개 단위로 개행 한번해라
          printf("\n");
      i++;
    }
    printf("\n");
    closedir(dp);
    return 0;
}

*/
/*

//ls -al 기능 구현1
int main(int argc,char **argv)
{
    int cmd;
    while((cmd=getopt(argc,argv,"ab"))>0)//getopt 는 a나 b옵션을 받는기능
                                    //argc 인자갯수 argv 파일=(문자열)의 주소
    {
      printf("cmd = %c\n",cmd);
      switch(cmd)
      {
          case 'a':

            printf("a option\n");
            break;

          case 'b':
            printf("b option\n");
            break;

          default:   //getopt  선언하지 않은것을 출력하면 ? 이 나옴.
            printf("unknown option\n");
      }
```

```
    }
    return 0;
}

*/

/*
//싹다 까발리는 옵션 숨긴 파일이던 뭐든
int main(int argc,char **argv)
{
    DIR *dp;
    int i=0,cmd;
    struct dirent *p;
    cmd=getopt(argc,argv,"a");//인자를 받는데 a만 받는다.
    dp=opendir(".");
    while(p=readdir(dp))//현재 리스트를 뽑아온다
    {
      if(cmd!='a')//a가 살아있지 않으면 .을 재끼겠다
      {
          if(p->d_name[0]=='.')//숨김 파일은 재낀다 .
            continue;
      }
      printf("%-16s ",p->d_name);//디렉토리 안에 있는 리스트를 출력.
      if((i+1)%5==0)//5행마다 개행
          printf("\n");
      i++;
    }
    printf("\n");
    closedir(dp);
    return 0;
}

*/
/*
int main(int argc,char **argv)
{
    DIR *dp;
    int i=0;
    int cmd;
    struct dirent *p;
    int flag=0;
    while((cmd=getopt(argc,argv,"alRi"))>0)
    {
      switch(cmd)
      {
          case 'a':
            flag |=1;   //flag 비트 연산인데 오른쪽서부터 저장된다고 생각하면 된다.
            break;      // -a 기능 2^1 이 저장?
          case 'l':
            flag |=2;  //     -l 기능 2^2 저장
            break;
          case 'R':
            flag |=4;   // -r 기능 2^4 저장
            break;
          case 'i':      //-i 기능 2^8 저장
            flag |=8;
            break;
      }
    }


    dp=opendir(".");
    while(p=readdir(dp))
```

```
    {
      if(!(flag&1))
      {
          if(p->d_name[0]=='.')
            continue;
      }
      printf("%-16s ",p->d_name);
      if((i+1)%5==0)
          printf("\n");
      i++;
    }
    printf("\n");
    closedir(dp);
    return 0;

}
*/
/*

int main(int argc,char **argv)
{
    struct stat buf;//stat 은 상태를 봄
    char ch;
    stat(argv[1],&buf);//처음 받은 인자를 읽어서 버프에 저장
    if(S_ISDIR(buf.st_mode))//디렉토리, 연산
      ch='d';
    if(S_ISREG(buf.st_mode))//파일
      ch='-';
    if(S_ISFIFO(buf.st_mode))//파이프
      ch='p';
    if(S_ISLNK(buf.st_mode))//링크 바로가기 같은거
      ch='l';
    if(S_ISSOCK(buf.st_mode))//소켓 네트워크 관련 장치 c나b로 대부분 되어있다
      ch='s';
    if(S_ISCHR(buf.st_mode))//캐릭터 디바이스
      ch='c';
    if(S_ISBLK(buf.st_mode))//블럭 디바이스
      ch='b';
    printf("%c\n",ch);
    return 0;
}

*/
/*
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))
      perm[0]='d';
    if(S_ISREG(buf.st_mode))
      perm[0]='-';
    if(S_ISFIFO(buf.st_mode))
      perm[0]='p';
```

```
    if(S_ISLNK(buf.st_mode))
      perm[0]='l';
    if(S_ISSOCK(buf.st_mode))
      perm[0]='s';
    if(S_ISCHR(buf.st_mode))//캐릭터 디바이스 = 논블럭 디바이스
      perm[0]='c';
    if(S_ISBLK(buf.st_mode))//블럭 디바이스
      perm[0]='b';
    for(i=0;i<9;i++)
      if((buf.st_mode>>(8-i))&1)//8~0 까지
          perm[i+1]=rwx[i%3];
    printf("%s ",perm);
    printf("%ld ",buf.st_nlink);//       ./a.out .  자기자신 ./a.out .. 상위
    pw=getpwuid(buf.st_uid);
    printf("%s ",pw->pw_name);
    gr=getgrgid(buf.st_gid);
    printf("%s",gr->gr_name);
    return 0;
}

*/


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))
               perm[0]='d';
        if(S_ISREG(buf.st_mode))
               perm[0]='-';
    if(S_ISFIFO(buf.st_mode))
         perm[0]='p';
     if(S_ISLNK(buf.st_mode))
         perm[0]='l';
     if(S_ISSOCK(buf.st_mode))
         perm[0]='s';
     if(S_ISCHR(buf.st_mode))
         perm[0]='c';
     if(S_ISBLK(buf.st_mode))
         perm[0]='b';
     for(i=0;i<9;i++)
       if((buf.st_mode>>(8-i))&1)//8진수 : 파일 권한 설정
          perm[i+1]=rwx[i%3];//3비트 rwx 표현

     for(i=0;i<3;i++)
       if((buf.st_mode>>(11-i))&1)
           if(perm[(i+1)*3]=='-')
             perm[(i+1)*3]=sst[i]^0x20;//대소문자 변경
           else
             perm[(i+1)*3]=sst[i];
//sst : t 공유파일
```

```
    printf("%s ",perm);
    printf("%ld ",buf.st_nlink);
    pw=getpwuid(buf.st_uid);
    printf("%s ",pw->pw_name);
    gr=getgrgid(buf.st_gid);
    printf("%s",gr->gr_name);
    printf("%d",buf.st_size);
    tm=localtime(&buf.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);
    printf("\n");
    return 0;
}
```