TI DSP, MCU 및 Xilinx Zynq FPGA 프로그래밍 전문가 과정

강사 - 이상훈
gcccompil3r@gmail.com
학생 – 이우석
colre99@naver.com

[3/22(목) – 21 일차]

리눅스 명령어 ls 는 파일의 상태 정보를 확인할 때 사용하는 명령어.
Ls -a 명령 결과의 가장 앞부분 10 자리는 파일 종류와 소유자의 권한(rwx)),
소유 그룹의 권한,GUEST 권한이다. 그리고, 나오는 숫자는 링크 수를 의미.
이어서 소유자와 소유자의 그룹이 나오고 다음의 수는 파일의 크기를 의미한다.
마지막으로는, 마지막 수정 시각과 파일명이 나타난다.

```
#include <sys/stat.h>
```

S_ISDIR – 디렉토리 파일인지 판별
S_ISREG – 정규 파일인지 판별
S_ISFIFO – FIFO 파일인지 판별
s_ISSOCK – 소켓 파일인지 판별
S_ISCHR – 문자 장치 파일인지 판별
S_ISBLK – 블록 장치 파일인지 판별

st_nlink : 파일의 하드링크 수
st_uid : user ID
st_gid : group ID
st_size : 파일의 사이즈
st_blocks : 파일에 할당한 블럭의 수

rwx = (읽고, 쓰고, 실행)

[struct stat]

맨앞의 4 비트는 파일의 종류를 나타내거나 구분할때 사용된다.
ex)0001 0000 0000 0000
1 을 디렉토리라고 가정할 경우 맨앞 4 비트로 파일이 디렉토리인지 파일인지 구분한다.
(권한을 지정할때는 8 진수를 이용한다.)

* chmod = change mode (셸 명령어)
이 명령어는 파일들이나 디렉토리의 파일 시스템 모드들을 바꾼다.ex)권한.
그 모드들은 허가나 특별한 모드들을 포함한다.

Chmod 명령은 파일의 접근 허가 모드를 설정할 때 사용한다.. 그리고 이 외에도 chmod 명령은 sticky bit 와 set user id bit 를 설정할 수 있다.

Sticky bit 를 설정한 프로그램 파일을 한 번 실행하면 사용한 프로그램의 Text 세그먼트 (코드를 기계어로 번역한 영역) 를 swap 파티션에 배치하여 빠르게 실행할수 있게 하였다. 하지만 가상 메모리 처리로 지금은 swap 파티션에 배치하지 않아도 속도 차이가 나지 않는다. 지금은 sticky bit 를 설정하면 사용 권한은 부여하지만 파일의 삭제를 하지 못하게 할 때 sticky bit 를 설정한다. 그리고 sticky bit 를 설정할 때는 t 옵션을 추가한다. Chmod + o + t temp 처럼 사용한다.

그리고 chmod 명령을 통해 프로그램 파일에 set user id bit 를 지정하면 실행할 때 프로세스는 로그인 한 실제 사용자 ID (real user id) 권한이 아닌 소유자 ID 권한으로 동작한다. 참고로 프로세스가 동작할때 유효 사용자 ID 권한 (effective user id) 권한으로 동작한다고 말한다.

ex) chmod 4644 = 대문자는 권한이 없다. ex) -rwSr—r-- 1
chmod 4755 = 소문자는 권한이 있다. ex) -rwxr-xr-t 1

chmod 0000 a.txt 하고 ls -l 하면 디렉토리 내에 파일들을 확인 할 수 있고, a.txt 의 파일 권한이 있는지 없는지 확인할수 있다.
(차이점 이라면 권한이 없을땐, 파일에 대문자가 껴있고, 권한이 있을땐 뒤로 빠져있다.)

[paging(페이징)] -응답속도를 빠르게 하기위해-

페이징 할때 우리가 필요한건 물리메모리에 들어있는 데이터.
하드디스크 물리메모리에 옮기고(=swap). Boot 가 있고, swap 은 메모리 사이즈의 두배.
가상을 물리메모리로 변환하는 과정을 페이징.

소프트 캐시를 도와주기 위해 도와주는것이 sticky bit. 페이지 자체를 swap 에다가
넣는다. Swap 이 LRU 매커니즘을 사용하며, 메모리 옮기는걸 도와주는게 swap.

최소단위: page *메모리를 옮기는 작업을 페이징 이라고 일컫음.

덧붙여, 가상에서 32bit 는 비율이 1:3 , 64bit 는 비율이 1:1.

Sticky bit = t (디렉토리에 t 가 붙으면 공유파일이 되고
파일에 붙으면 swap 을 할지말지 정하는것.)
sticky bit 를 set 할거면 chmod 명령어를 사용(swaping)

1.

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

   if(S_ISDIR(buf.st_mode)) //directory냐고 묻는것. ex)0001
         perm[0] = 'd';
   if(S_ISREG(buf.st_mode)) // 레귤러. 일반 파일.
         perm[0] = '-';
   if(S_ISFIFO(buf.st_mode)) // 파이프. 0011
         perm[0] = 'p';
   if(S_ISLNK(buf.st_mode)) // 바로가기 파일 같은거.
         perm[0] = 'l';
   if(S_ISSOCK(buf.st_mode)) // 소켓.
         perm[0] = 's';
   if(S_ISCHR(buf.st_mode)) // 캐릭터 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) /* 오른쪽부터 채워져서
                        8 - 0 = 8(오른쪽)부터 8번째 = 즉, 첫번째자리 */
                     // 1 & 1 = 1(참) 이니깐 perm[0+1] = rwx[0%3]=0
                     // 그럼 1번째 자리에 0(읽다)r.
         perm[i + 1] = rwx[i % 3];
   printf("%s ", perm); //입력 들어온 파일에 대한 출력.
   printf("%lu ", buf.st_nlink); //파일 개수 알아보는거.nlink
   pw = getpwuid(buf.st_uid); //시스템콜 .. 유저 아이디 가져오기
   printf("%s ", pw->pw_name);
   gr = getgrgid(buf.st_gid); // 그룹아이디 가져오기
   printf("%s\n", gr->gr_name);

   return 0;
}
```

결과값:

```
wooseok91@air:~/my_proj/Homework/sanghoonlee/lec/lsp$ gcc ls2.c
wooseok91@air:~/my_proj/Homework/sanghoonlee/lec/lsp$ ./a.out a.out
-rwxrwxr-x- 1 wooseok91 wooseok91
```

2.

```
16
17    char ch;
18    char perm[11] = "----------";
19    char rwx[4] = "rwx";
20    char sst[4] = "sst";
21                              //위에보면 struct stat buf 라는 변수선언
22    int i;                    //buf라는 구조체에서 st_mode를 가져온다.
23    stat(argv[1], &buf);      //S_ISXXX 이게 함수.
24
25 if(S_ISDIR(buf.st_mode))  //. 구조체안에 st_mode를 가져온다.
26    perm[0] = 'd';
27 if(S_ISREG(buf.st_mode))  // is reg ?라는뜻. 참이면 1(실행) , 거짓이면 0
28    perm[0] = '-';
29 if(S_ISFIFO(buf.st_mode)) // 참이면 밑에 perm[0] = '*'; 실행. 즉, p
30    perm[0] = 'p';
31 if(S_ISSOCK(buf.st_mode))
32    perm[0] = 's';
33 if(S_ISCHR(buf.st_mode))
34    perm[0] = 'c';                  //st_mode : 파일의 모드를 다룬다.
35 if(S_ISBLK(buf.st_mode))
36    perm[0] = 'b';
37 for(i = 0; i < 9; i++)           // rwx 실행 끝나고 돈다.
38       if((buf.st_mode >> (8 - i)) & 1)
39          perm[i + 1] = rwx[i % 3];  // 여기 for문까지 끝내고 다음 for문 시작.
40 for(i = 0; i < 3; i++)          // 앞 3자리 추가.
41       if((buf.st_mode >> (11 - i)) & 1)
42          if(perm[(i + 1) * 3] == '-') // -(작대기)자체가 권한이 없다는 뜻.
43          perm[(i + 1) * 3] = sst[i] ^ 0x20; //xor 0x20 = 32 = 대소문자.
44       else                              //권한이 있으면, 대소문자 바꾸고
45          perm[(i + 3) * 3] = sst[i];   // 없으면 sst 실행.
46                    // => execute = x. 랑 같다는 말이며 권한이 있다는거.
47                                 //set uid = root의권한을 일시적 갖음.
48 printf("%s", perm);
49 printf("%ld", buf.st_nlink); //파일의 하드링크수
50 pw = getpwuid(buf.st_uid);   //유저 아이디
51 printf("%s", pw -> pw_name); //
52 gr = getgrgid(buf.st_gid);   //그룹 아이디
53 printf("%s", gr -> gr_name); //그룹 이름
54 printf("%lu", buf.st_size);  //파일의 사이즈
55 tm = localtime(&buf.st_mtime); //파일의 시간
56 printf("%d-%02d-%02d %02d:%02d",
57 tm -> tm_year + 1900, tm -> tm_mon + 1, tm -> tm_mday, tm -> tm_hour,
58 tm -> tm_min);
59 printf("\n");
60
61 return 0;
62
63 }
```

결과값 :

```
wooseok91@air:~/my_proj/Homework/sanghoonlee/lec/lsp$ ./a.out a.out
-rwxrwxr-x-1wooseok91wooseok9189842018-03-23 03:40
wooseok91@air:~/my_proj/Homework/sanghoonlee/lec/lsp$ ls -l
total 48
-rwxrwxr-x 1 wooseok91 wooseok91 8984  3월 23 03:40 a.out
-rwsr-xr-x 1 wooseok91 wooseok91    6  3월 20 17:49 a.txt
-rw-rw-r-- 1 wooseok91 wooseok91  686  3월 22 12:06 getopt1.c
-rw-rw-r-- 1 wooseok91 wooseok91 1325  3월 22 11:42 getopt.c
-rw-rw-r-- 1 wooseok91 wooseok91 1301  3월 22 14:41 ls1.c
-rw-rw-r-- 1 wooseok91 wooseok91 1509  3월 22 15:09 ls2.c
-rw-rw-r-- 1 wooseok91 wooseok91 2325  3월 22 21:53 ls3.c
-rw-rw-r-- 1 wooseok91 wooseok91  692  3월 22 13:35 ls.c
-rw-rw-r-- 1 wooseok91 wooseok91  659  3월 20 17:49 mycp.c
-rw-rw-r-- 1 wooseok91 wooseok91    6  3월 20 17:49 test.txt
wooseok91@air:~/my_proj/Homework/sanghoonlee/lec/lsp$ chmod 4644 a.txt
wooseok91@air:~/my_proj/Homework/sanghoonlee/lec/lsp$ ls -l
total 48
-rwxrwxr-x 1 wooseok91 wooseok91 8984  3월 23 03:40 a.out
-rwSr--r-- 1 wooseok91 wooseok91    6  3월 20 17:49 a.txt
```