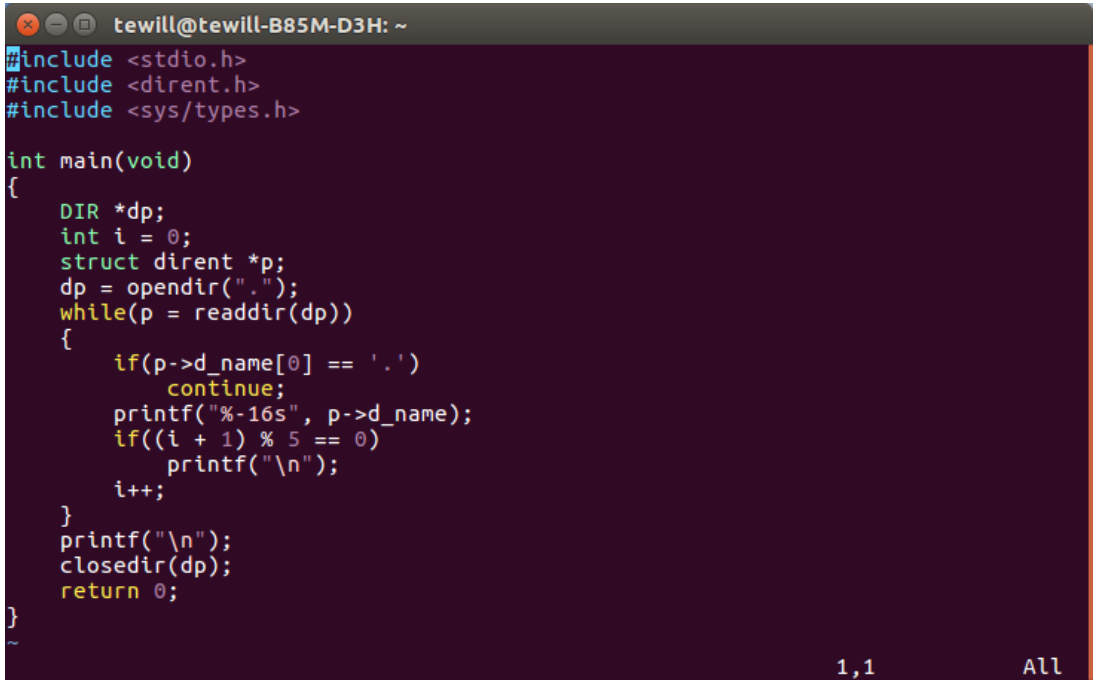
TI DSP, MCU 및 Xilinx Zynq FPGA 프로그램 전문가 과정

강사 - Innova Lee(이상훈)
gcccompil3r@gmail.com
학생 - 은태영
zero_bird@naver.com

기타

- ls 옵션 기능
- -a : .(자기 자신) / ..(상위 폴더) 표시 제거
- -R : 디렉토리 내부까지 전체 출력
- -i : inode 번호
- -l : 상세 정보

파일_ls



- opendir() : 디렉토리를 연다.
- 리턴 값이 배열 형식의 주소이다.

- “.” 을 통해 현재 위치의 디렉토리를 연다.
- 반복문을 통해 디렉토리 내부 파일들을 접근한다.
- 파일의 앞 문자가 . 일 경우 넘긴다.
- 파일 5개 단위로 ₩n 을 실행한다.

파일_ls

tewill@tewill-B85M-D3H: ~

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

int main(int argc, char **argv)
{
    int cmd;
    while((cmd = getopt(argc, argv, "ab")) > 0)
    {
        switch(cmd)
        {
            case 'a':
                printf("a option\n");
                break;
            case 'b':
                printf("b option\n");
                break;
            default:
                printf("unknown option\n");
        }
    }
    return 0;
}
~
                                                    1,1           All
```

- ❖ getopt(int, char *, char *) : 인자를 받아와 '-' 이후, 단어들을 리턴 한다.
- ❖ 반복문을 통해 옵션들을 받아온다.
- ❖ 반복문 안에서 받아온 리턴에 대해 switc를 통해 옵션에 따라 출력한다.

파일_ls

tewill@tewill-B85M-D3H: ~

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

int main(int argc, char **argv)
{
    DIR *dp;
    int i = 0, cmd;
    struct dirent *p;
    cmd = getopt(argc, argv, "a");
    dp = opendir(".");
    while(p = readdir(dp))
    {
        if(cmd != 'a')
        {
            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;
}
```

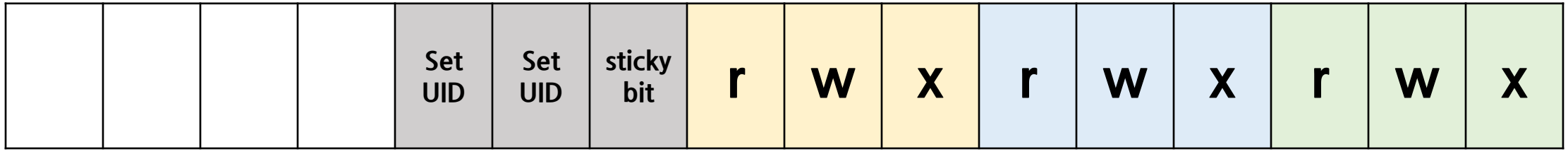
1,1 All

- ❖ 인자를 통해 옵션을 받아온다.
- ❖ 디렉토리를 열고, 반복문을 통해 파일들을 읽는다.
- ❖ 옵션을 확인하고, a 가 없을 경우 . / .. 을 넘긴다.
- ❖ p를 통해 파일의 이름을 출력한다.
- ❖ 5개 단위로 ₩n을 실행한다.

파일_ls

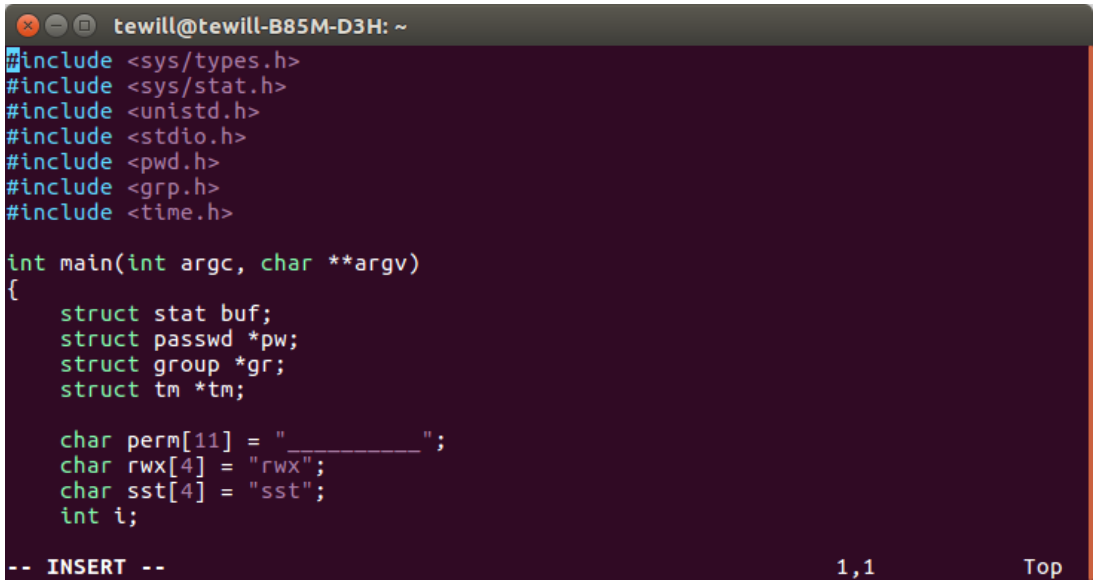
- 비트 단위로 옵션을 저장한다.
- |= 로 옵션을 저장하기 때문에 다른 옵션도 유지된다.
- 옵션 확인 방법으로 & 를 이용해 옵션을 확인한다.
- 비트 단위로 저장하기에, 2의 제곱으로 저장된다.
- 복수의 옵션에 대한 결과는 (00 & 00 | 00) 을 이용한다.

파일 권한



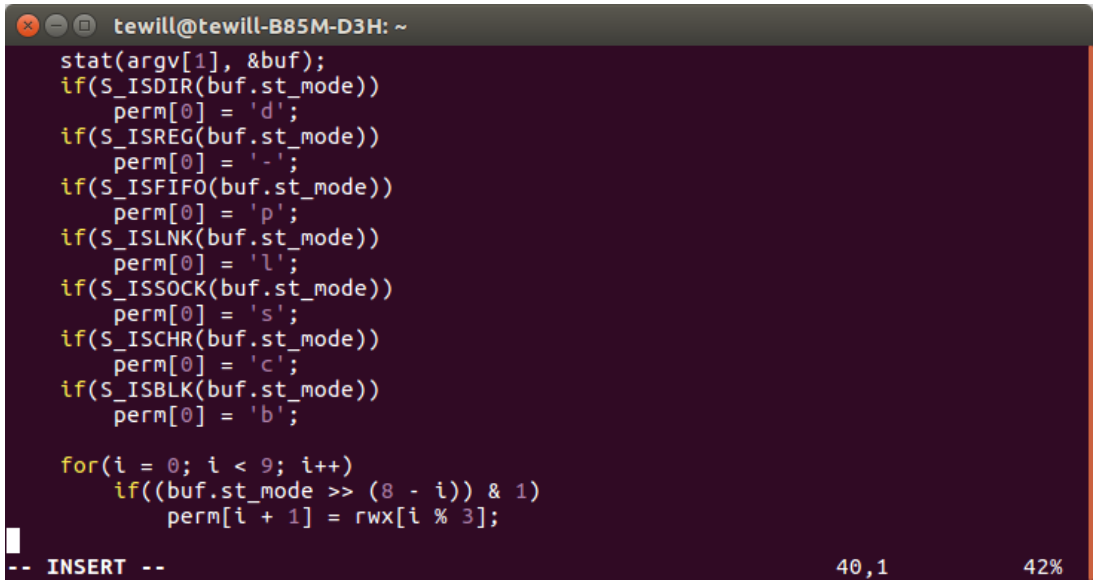
- ❖ 파일 종류[4bit] / 일시적 권한 부여[3bit] / 사용자 권한[3bit] / 그룹 권한[3bit] / 제 3자의 권한[3bit]
- ❖ 파일 권한 설정[RWX] : 8진수. r(Read) : 읽기 권한 / w(write) : 쓰기 권한 / x(execute) : 실행 권한
- ❖ SSS : SETUID : 사용자 권한 root 일시 부여 / SETUID : 그룹 권한 일시 부여(root or 소유자 권한) / sticky bit : 디렉토리 일 경우 공유 폴더, 일반일 경우 스왑(물리 메모리 백업) 지원.

파일_ls



- ❖ 구조체를 통해 사용자와 그룹, 시간을 만든다.
- ❖ 권한을 쓰기위해 rwx 문자열을 만든다.
- ❖ SSS 에 대해 쓰기 위해 sst 문자열을 만든다.
- ❖ 파일 권한을 쓰기 위해 perm 문자열을 만든다.

파일_ls



- 파일 타입에 따라 & 연산을 통해 구분한다.
- DIR : 디렉토리 파일.
- REG : 레귤러. 일반 파일.
- FIFO : 파이프 파일.
- LINK : 링크. 바로가기 파일.
- SOCK : 소켓 파일.
- CHR : 캐릭터 디바이스.
- BLK : 블럭 디바이스.
- 파일의 정보를 받아 비트 연산을 통해, 권한을 perm 문자열에 입력한다.

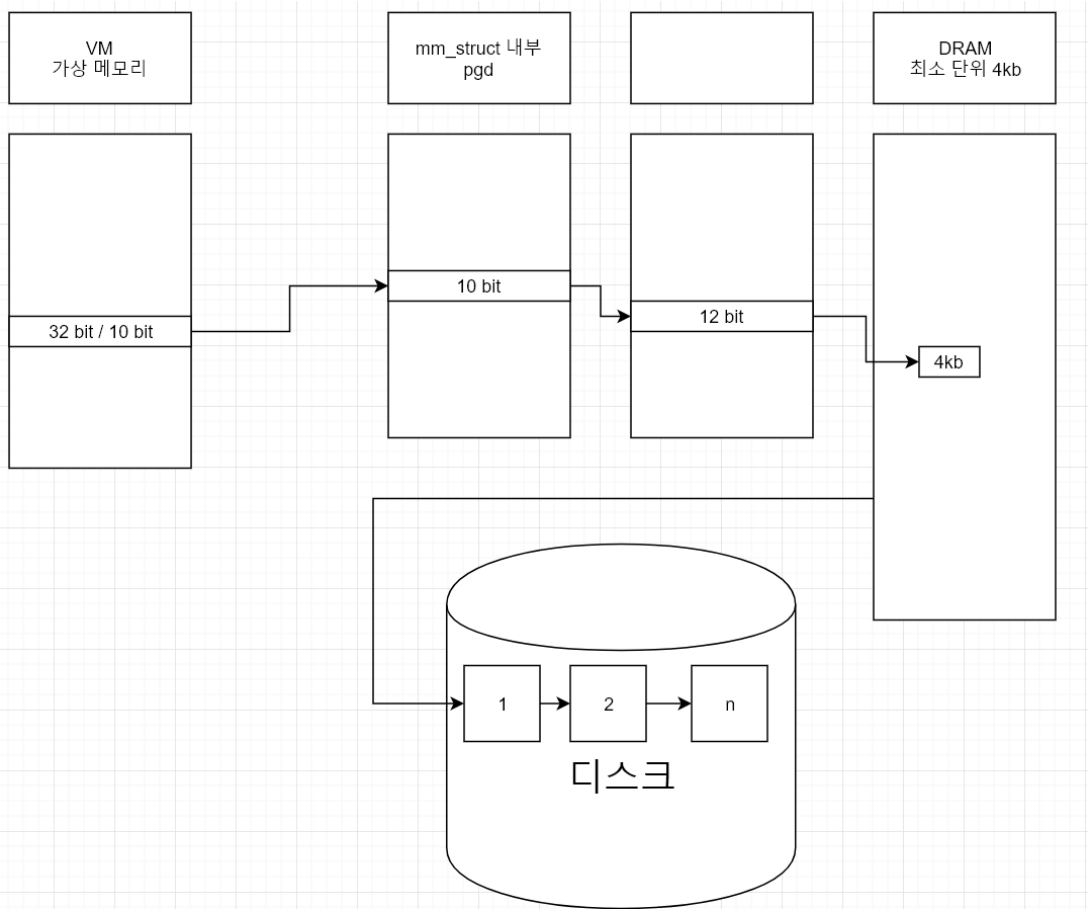
파일_ls

tewill@tewill-B85M-D3H: ~

```
    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;
            else
                perm[(i + 1) * 3] = sst[i];



    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("%ld ", 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;
}
-- INSERT --                                          67,1          Bot
```

- SSS 처리를 하기 위해 해당 정보를 읽어서, 참 일 경우 perm 의 3, 6, 9 번째가 '-' 일 때 대문자 S ,S , T 를 아닐 경우 소문자 s, s, t 를 입력한다.
- 첫 출력은 파일의 종류와 권한을 나타낸다.
- 두 번째 출력은 디렉토리 숫자를 나타낸다.
- 세 번째 출력은 유저 ID 를 나타낸다.
- 네 번째 출력은 그룹 ID 를 나타낸다.
- 다섯 번째 출력은 사이즈를 나타낸다.
- 마지막 출력은 생성 시간을 나타낸다.

Paging_Swap



- Paging : 가상 메모리에서 물리메모리의 주소를 찾아가는 과정 이다.
- 가상 메모리의 주소를 32bit 기준 10bit / 10bit / 12bit 로 구분하고, 그것을 통해 배열의 인덱스를 찾는다.
- Demand On Paging : 지금 필요한 것만 Paging 하는 것을 말한다.
- Swap : 메모리로 커버가 안될 경우, 디스크에 백업을 하여 정보를 불러오는 것을 말한다.
- LRU 알고리즘 : 상대적으로 안 쓰는 것을 Swap 하도록 하는 알고리즘이다.