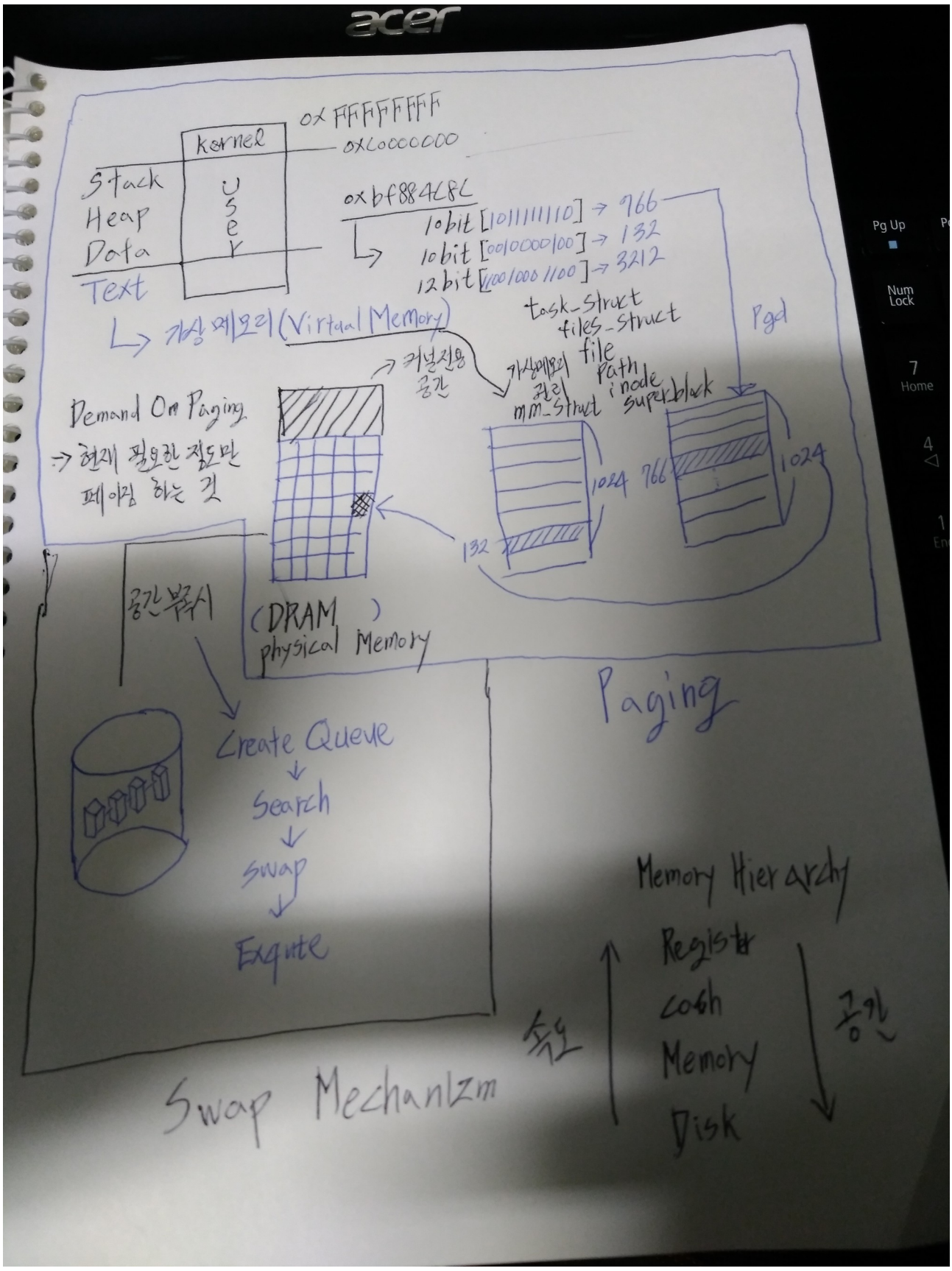
TI DSP, MCU 및 Xilinx Zynq FPGA 프로그래밍 전문가 과정

2018.03.22
21 일차

강사 – Innova Lee(이상훈)
gcccompil3r@gmail.com

학생 – 신민철
akrn33@naver.com

가상메모리는 물리메모리의 가상으로서역할을 하는데,
프로세스당 하나의 가상메모리가 주어지며 가상메모리는
3 개로 쪼개지며 첫 번째로 쪼개어진 비트가 배열의 주소를 두번째로 쪼개진 비트가 갖고,
두번째로 쪼개진 배열의 주소를 세번째로 쪼개진 비트가 가진다. 세번째로 쪼개진 비트는 실제 메모리 주소를 가리키고 나머지 첫번째 두번째로 쪼개진 비트들은 가상의 주소를 가진 배열을 가짐으로써 부족한 메모리의 공간을 효율적으로 사용할 수 있게 된다. 이 가상메모리들은 파일구조체 안의 mm_struct 라는 구조체가 관리한다. 레드블랙트리로 이루어져 있다고 한다. 프로그램 용량이 커서 메모리가 전부 할당되고 다른 프로그램이 돌아가지 않으면 안되니까 이런 문제를 해결하기 위해서 디스크 swap 메커니즘이 생긴다. swap 메커니즘은 디스크 안에 메모리에서 필요하긴 하지만 항상 쓰지 않는 프로세스를 디스크에 저장시켰다가 사용할 때마다 꺼내서 쓰는 방식이다. 이와 동시에 디스크 안에서 LRU 알고리즘이 적용되어 이 안에서도 관리를 하게된다. 이렇게 되면 효율적으로 관리가 가능하게 된다.
속도가 레지스터보다는 느리지만 빠른처리속도를 가진 메모리는 저장공간이 부족할 수 있는데 이를 보완한 것이 페이징 기법인 것 같다.


0xFFFFFFFF
0xC0000000
kernel
Stack
Heap
Data
User
Text
0xbf884c8c
10bit [1011111110] → 766
10bit [0010000100] → 132
12bit [1000001100] → 3212
가상메모리 (Virtual Memory)
커널전용 공간
task_struct
files_struct
file
Path
inode
superblock
가상메모리 관리 mm_struct
Pgd
Demand On Paging
→ 현재 필요한 것만 페이징 하는 것
1024
766
132
1024
공간부족
(DRAM)
physical Memory
Paging
Create Queue
Search
Swap
Exqute
Swap Mechanism
Memory Hierarchy
Register
cash
Memory
Disk
속도
용량

lsmo.c

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

int main(void)
{
	DIR* dp;//파일구조체와 비슷한데 얘는 디렉토리임
	int i = 0;
	struct dirent *p;//디렉토리 내의 리스트
	dp = opendir(".");//디렉토리도 열고쓰려면 오픈해야함 현재를 나타내는 .을 사용
	while(p = readdir(dp))//포인터를 읽어냄
	{
		if(p->d_name[0] =='.')//내용물들이 리스트형태로 존재맨앞의 글자가 .이면 재끼고
			continue;
		printf("%-16s",p->d_name);//.이아니면 출력
		if((i+1)%5 ==0)//5 개 넣으면 개행해라
			printf("\n");
		i++;
	}
	printf("\n");//다 출력하고나서 개행
	closedir(dp);//오픈했으니 클로즈해서 쌍을 맞춰준다.
	return 0;
}

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

```
int main(void){
	int fd, ret;
	char buf[1024];
	mkfifo("myfifo");//fifo 통신파일생성
	printf("%d",mkfifo("mkfifo"));
	fd = open("myfifo", O_RDWR);//fifo 통신에 입력
	for(;;)
	{
		ret = read(0, buf, sizeof(buf));
		buf[ret - 1] = 0;
		printf("Keyboard Input : [%s]\n",buf);
		read(fd, buf, sizeof(buf));
		buf[ret -1] = 0;
		printf("Pipe Input : [%s]\n",buf);
	}
	return 0;
}
```

lsmo2.c

```
#include<stdio.h>
#include<fcntl.h>
#include<unistd.h>
int main(int argc, char* argv[])
{
	int cmd;
	while((cmd = getopt(argc, argv, "ab")) > 0)//ab 는 처리할 수 있는 옵션이고 몇개가 왔는지 감지하기 위해서 argc 와 argv 를 넣은거다.
	{
		switch(cmd)
		{
			case 'a':
				printf("a option\n");
				break;
```

```
                case 'b':
                        printf("b option\n");
                        break;
                default:
                        printf("unknown option\n");
        }

    }
    return 0;
}
```

lsmo3.c

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

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

lsmo7.c

```
#include<stdio.h>
#include<fcntl.h>
#include<stdlib.h>
#include<unistd.h>
#include<grp.h>
#include<pwd.h>
#include<sys/types.h>
#include<sys/stat.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))//디렉토리이면 d
            perm[0] = 'd';
        if(S_ISREG(buf.st_mode))//일반파일이면 -
            perm[0] = '-';
```

```
        if(S_ISFIFO(buf.st_mode))//파이프통신 파일이면 p
                perm[0] = 'p';
        if(S_ISLNK(buf.st_mode))//바로가기 파일이면 l
                perm[0] = 'l';
        if(S_ISSOCK(buf.st_mode))//네트워크 파일이면 s
                perm[0] = 's';
        if(S_ISCHR(buf.st_mode))//캐릭터파일이면 c
                perm[0] = 'c';
        if(S_ISBLK(buf.st_mode))//블록파일이면 b
                perm[0] = 'b';
        for(i = 0; i < 9; i++)
        {
                if((buf.st_mode >> (8-i) &1))
                        perm[i +1] = rwx[i % 3];
                printf("%s",perm);
                printf("%d",buf.st_nlink);
                pw = getpwuid(buf.st_uid);
                printf("%s",pw->pw_name);
                gr = getgrgid(buf.st_gid);
                printf("%s",gr->gr_name);
        }
        return 0;
}
```