1. 요구분석 (계획)

◉ 플래시 메모리 특징

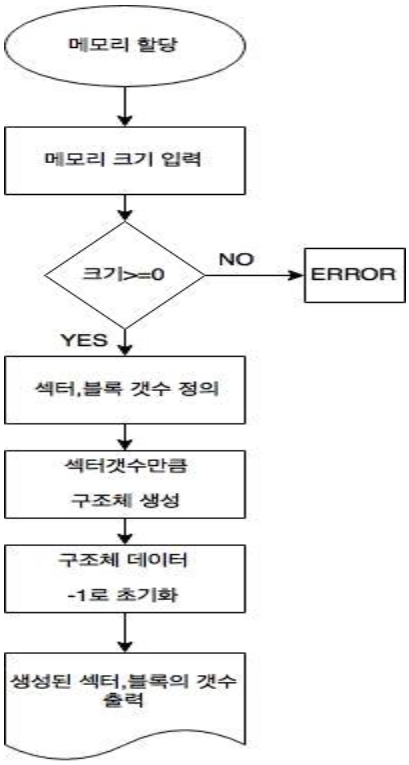
플래시 메모리는 erase-before-wirte 라는 특징이 있는데 이것은 데이터를 업데이트 하기위해서는 그 장소가 비워져 있어야 하며, 데이터가 있는 경우 쓰여지기 전에 지워져야한다는 특성이다. 지우기 위한 메모리는 읽기, 쓰기에 비해 크기 때문에 지우기 연산이 많아질수록 성능이 저하되는 단점이 있다.

◉ 과제 설명

본 과제는 메모리의 크기를 사용자로부터 입력받아 WRITE,READ,ERASE 명령을 수행한다.
1섹터의 크기는 512byte 이고 1블록은 32섹터로 구성된다.
초기에 데이터는 “-1” 로 초기화하고 명령을 수행한다.
WRITE 명령 수행시 데이터가 있는 위치에 데이터를 쓰면 Overwrite Error 가 발생한다.
ERASE 명령은 섹터단위가 아닌 블록단위로 지우기를 수행한다.

○ 메모리 할당

FlowChart



설명

●메모리 할당

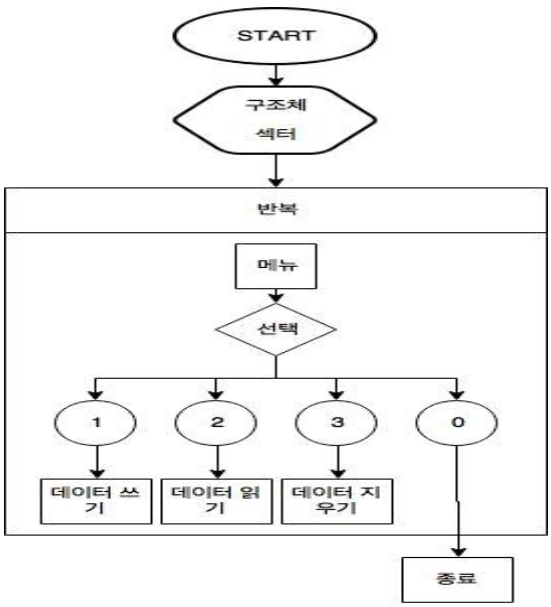
●크기가 0보다 작으면 에러

●섹터는 512byte
●블록은 32개의 섹터
●1섹터=n * 1024 * 1024 / 512

●섹터,블록의 개수 출력

○ 기능 선택

FlowChart



설명

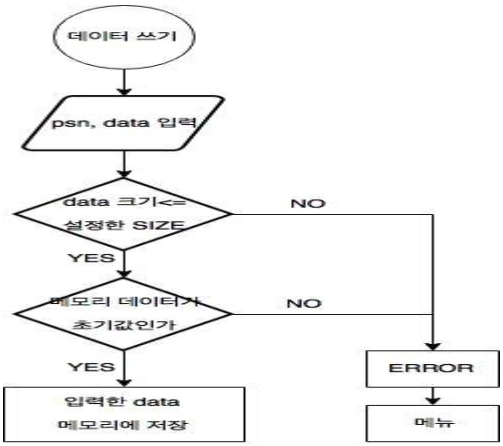
●시작

●1.데이터 쓰기
●2.데이터 읽기
●3.데이터 지우기

●0.종료

○ 데이터 쓰기

FlowChart



설명

●물리주소, 데이터 입력

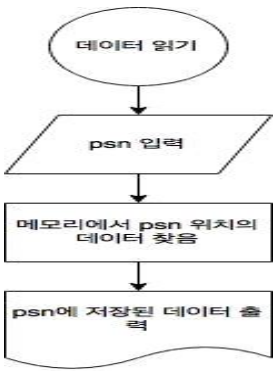
●데이터 배열 길이 초과 에러

●메모리에 데이터가 있으면 overwrite 에러

●데이터가 없으면 저장

○ 데이터 읽기

FlowChart



설명

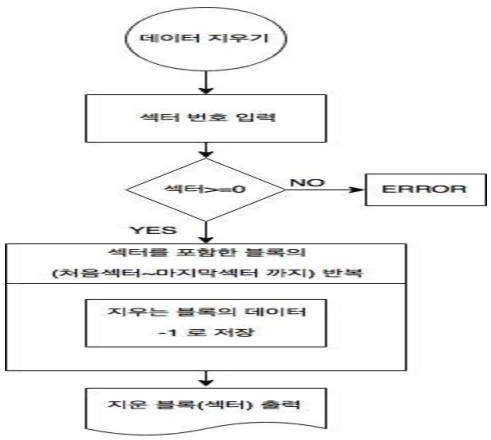
● 읽어올 물리주소 입력

●물리주소 위치의 데이터 찾음

●저장된 데이터 출력

○ 데이터 지우기

FlowChart



설명

● 지울 섹터 번호 입력

●섹터번호가 0보다 작으면 에러

●입력한 섹터번호의 블록을 찾아서 블록전체를 지움

●저장된 데이터 출력

○주요 함수 기능 및 설명

○ 메모리 생성 / 초기화

```
sector = n * 1024 * 1024 / 512; //섹터의 갯수
block = sector / 32; //블록의 갯수 (1블록은 32섹터)

memory = (MEMORY*)malloc(sizeof(MEMORY)*(sector)); //입력받은

for (i = 0; i < sector; i++)
{
    strcpy(memory[i].memory,"-1"); // 데이터 -1 로 초기화
}

for (i = 0; i < sector; i++)
{
    printf("psn%d = %s₩n", i,memory[i].memory);
}
```

1섹터는 512바이트로 가정하여 구조체에 char형배열 512 만큼 설정하고
입력받는 단위가 MB이므로 섹터 개수를 결정한다
섹터 개수만큼 구조체를 생성하고 데이터를 -1로 초기화한다.

○ 데이터 쓰기

```
if (strlen(data) >= SIZE) //입력한 데이터크기 예외처리
{
    printf("data size ERROR!!₩n");
    return 0;
}

if (!(strcmp(memory[psn].memory, "-1") == 0)) //데이터에 초기값이 아닌 다른 데이터가 있을때
{
    printf("Overwrite ERROR!!₩n"); //에러
    return 0;
}

strcpy(memory[psn].memory, data);
```

데이터를 입력받아 저장된 메모리에 데이터가 없을 때 입력한 데이터를 저장한다.
메모리에 데이터가 있는 상태라면 Overwrite 에러를 출력한다.

○ 데이터 읽기

```
void Flash_read(MEMORY *memory) // 읽기
{
    int psn;

    printf("psn:");
    scanf("%d", &psn); //물리주소 입력

    printf("data : %s ₩n", memory[psn].memory);
    printf("₩n");
}
```

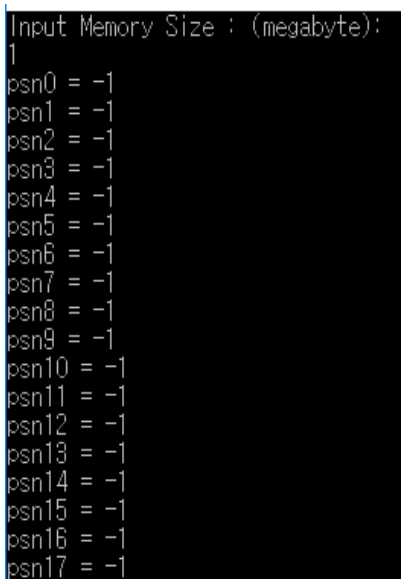
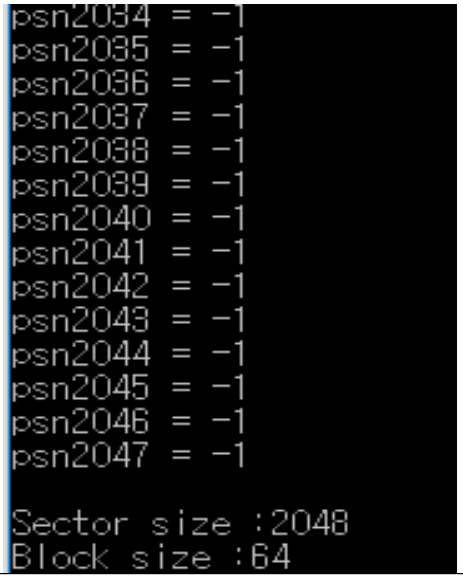
물리주소를 입력받아 메모리에서 물리주소를 찾아 그 위치에 저장된 데이터를 출력한다.

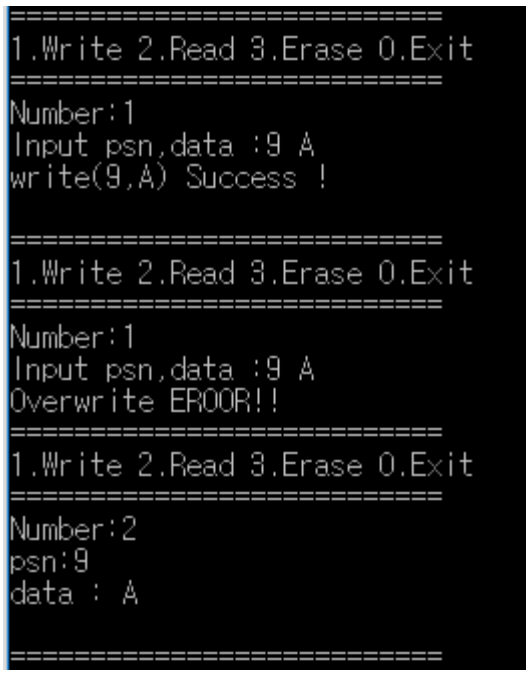
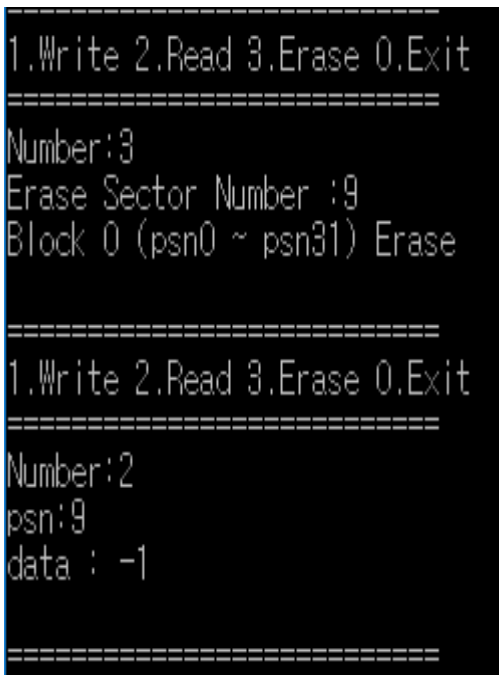
○ 데이터 지우기

```
void Flash_erase(MEMORY *memory) // 지우기(블록단위)
{
    int sector;
    int i;
    printf("Erase Sector Number :");
    scanf("%d", &sector);

    if (sector <= 0)
    {
        printf("ERROR!!₩n");
        return 0;
    }

    for (i = (sector / 32) * 32; i <= ((sector/32)*32)+32; i++) //블록단위로 데이터 -1 로 초기
    {
        strcpy(memory[i].memory, "-1");
    }
```

지우고자 하는 섹터 번호를 입력받아서 그위치에 있는 데이터를 -1로 초기화한다.
단, 지우기 연산은 블록단위로 지워지기 때문에 입력받은 섹터를 포함한 블록의 전체 섹터를 -1로 초기화하여야 한다.

○ 테스트 결과

○데이터 쓰기(overwrite) / 읽기	○데이터 삭제
	

○데이터 쓰기(overwrite) / 읽기	○데이터 삭제 / 삭제후 확인
	

○ 결과분석 및 문제점

- 요구분석 / 계획에서 이행하지 못한 점 없음.