P2 두 개 이상의 장치 다루기 결과보고서

3조 20162581 손승표

"dot.c" 파일의 빈칸을 채워넣어 소스코드를 완성하고, 다음의 질문에 답해보자.

```
//-------| src/dot.c |-------//
#include "dot.h"


static unsigned short dot_decimal[10][MAX_DOT] = {
 {0x7f, 0x41, 0x41, 0x41, 0x7F}, // 0
 {0x00, 0x00, 0x7F, 0x00, 0x00}, // 1
 {0x4f, 0x49, 0x49, 0x49, 0x79}, // 2
 {0x49, 0x49, 0x49, 0x49, 0x7f}, // 3
 {0x78, 0x08, 0x08, 0x7f, 0x08}, // 4
 {0x79, 0x49, 0x49, 0x49, 0x4f}, // 5
 {0x7f, 0x49, 0x49, 0x49, 0x4f}, // 6
 {0x40, 0x40, 0x40, 0x40, 0x7f}, // 7
 {0x7f, 0x49, 0x49, 0x49, 0x7f}, // 8
 {0x78, 0x48, 0x48, 0x48, 0x7f}  // 9
};

static short * dot[MAX_DOT];

void init_dot(short * address[]) {
 int i;
 for( i=0; i<MAX_DOT; i++ ) {
  dot[i] = address[i];
 }
}

void dot_clear() {
 int i;
 for(i=0; i<MAX_DOT; i++){
  *dot[i] = 0;
 }
 usleep(0); // for Ximulator
}

void dot_write(int decimal) {
 int i;
 for(i=0; i<MAX_DOT; i++){
  *dot[i] = dot_decimal[decimal][i];
 }
 usleep(0); // for Ximulator
}
```

```
void dot_up_shift(int decimal) {
 int shift, i;
 for( shift=0; shift<=7; shift++ ) {
  for(i=0; i<MAX_DOT; i++){
   *dot[i] = dot_decimal[decimal][i] << shift;
  }
  usleep(50000);
 }
}

void dot_down_shift(int decimal) {
    int shift, i;
    for( shift=0; shift<=7; shift++ ) {
        for(i=0; i<MAX_DOT; i++){
            *dot[i] = dot_decimal[decimal][i] >> shift;
        }
        usleep(50000);
    }
}

void dot_inverse(int decimal) {
 int shift, i;
 for( shift=0; shift<3; shift++ ) {
  for(i=0; i<MAX_DOT; i++){
   *dot[i] = dot_decimal[decimal][i];
  }
  usleep(300000);
        for(i=0; i<MAX_DOT; i++){
            *dot[i] = ~dot_decimal[decimal][i];
        }
 }
 dot_clear();
}
```

a. main.c의 각 함수가 하는 역할에 대하여 설명하여라.

main.c 에는 `main, mapper, unmapper, emergency_closer` 4개의 함수가 있다.

- int main()
  - 프로그램의 주요 동작이 시작돼서 끝나는 함수
- short* mapper()
  - led 장비를 메모리에 매핑해 그 주소값을 가져오는 함수.
  - 해당 주소값으로 led를 동작할 수 있다.
- void unmapper()
  - 매핑한 led 메모리를 해제하는 함수
- void emrgency_closer()
  - 긴급히 종료되어야할 경우 부르는 함수
  - 매핑 해제하고 fd도 닫으며, 프로그램을 종료시킨다.

b. 프로그램을 실행하고 3, 26, 32, 45를 입력할 때, 각각 어떤 결과가 예상되는가?

- 3
  - led에 숫자 3이 들어온다.
- 26
  - led에 숫자 6이 들어오며,
  - dot 불이 내려가며 꺼진다.
- 32
  - led에 숫자 2가 들어오는 것과
  - 2에 반전되는 영역의 불이 들어오는 것이 반복된다.
- 45
  - 프로그램이 종료된다.

c. 만약 Shift연산을 사용할 수 없다면, 어떤 연산을 활용하여 같은 효과를 낼 수 있는가? 왼쪽이나 오른쪽으로 1bit씩 이동하는 경우에 대해 각각 설명하여라.

- 숫자 2를 곱하는 것으로 활용할 수 있다.
- << 1 의 경우 전체에 2를 곱해주는 것과 같은 효과이다.
  - << n의 경우 2를 n번 곱해주는 것과 같기에 for문을 이용한다.

```
int result = 0xff;
for(int i = 0; i < n; i++)
{
    result *= 2;
}
```

- >> 1 의 경우 전체에 2를 나눠주는 것과 같은 효과이다.
  - >> n의 경우 2를 n번 나눠주는 것과 같기에 for문을 이용한다.
  - 하지만 대용 for문 코드는 signed의 경우에서 부호를 복사하지 않기에 주의해야한다.

```
int result = 0xff;
for(int i = 0; i < n; i++)
{
    result /= 2;
}
```