1. 응용프로그램 작성

1-1. LCD에 Colorbar 표시하기

학습목표	● LCD에 colorbar 그리는 응용프로그램을 작성한다.

수행 내용 / LCD에 Colorbar 표시하는 응용프로그램 작성

기기

- 호스트 PC, 타겟 장비, USB 메모리, USB to Serial Cable, USB 메모리 스틱

안전/유의사항

- 장비와 호스트 PC가 네트워크 망에 연결되어 있으면 네트워크를 통해 파일을 타겟 장비에 전송할 수 없으므로 USB 메모리나 minicom의 zmodem을 이용하여 전송한다.

수행/순서

(1) 개요

이장에서 프레임버퍼 디바이스 드라이버를 사용하여 LCD에 칼라바를 로딩하는 프로그램을 작성해 본다.

프레임 버퍼(Frame Buffer)은 커널에서 한 화면을 구성하기 위해서 할당 해 놓은 메모리라 할수 있다. 그래픽 하드웨어를 사용자 레벨의 응용 프로그램이 제어할 수 있도록 만들어진 디바이스 드라이버를 프레임 버퍼 드라이버라고 한다.

응용 프로그램을 통해 각 픽셀의 색상을 프레임 버퍼 드라이버에 전달하면 프레임 버퍼를 통해 LCD 화면에 표현하게 된다.

(2) 프로그램 작성

Color Bar를 LCD에 표현하는 것이다. Color Bar 신호는 일반적으로 일관된 색 재현(Consistent color reproduction)을 유지하기 위하여 사용된다. 공중파 방송에서 방송 전과 후, 화면조정 시간에 몇 가지 색상으로 화면을 구성하고 있는 것에서도 볼 수 있다. Color Bar는 몇 가지 종류가 있으며, 보통 7개의 수직 바가 있는데, 가장 왼쪽에 하나의 흰색 바가 있고, 이어서 오른쪽으로 노랑, 시안(cyan), 초록, 마젠타(magenta), 빨강 청색이다. 이것은 각 색의 휘도 레벨이 낮아지는 순서이고, 각 색은 적, 녹, 청의 3원색을 동일하게 7가지 색으로 배합한 것이다.

Color Bar를 LCD에 표현하는 예제를 통해 프레임 버퍼를 어떻게 다루는 지 실습해 보도록 한다.

① 이미 작성된 소스 파일은 첨부 파일을 참조한다. 호스트 PC에서 coloarbar.c 파일을 만들고 다음과 같이 작성한다.

colorbar.c

```
#include <stdio.h>
#include <stdlib.h>        // for exit
#include <unistd.h>        // for open/close
#include <fcntl.h>         // for O_RDWR
#include <sys/ioctl.h>     // for ioctl
#include <sys/mman.h>
#include <linux/fb.h>      // for fb_var_screeninfo, FBIOGET_VSCREENINFO

#define FBDEV_FILE "/dev/fb0"

int main (int argc, char **argv)
{
    int screen_width;
    int screen_height;
    int bits_per_pixel;
    int line_length;

    int fb_fd;
```

```
struct   fb_var_screeninfo fbvar; // 스크린의 정보를 획득하기 위한
struct   fb_fix_screeninfo fbfix;
unsigned char     *fb_mapped;
int mem_size;
unsigned long     *ptr;
int coor_y;
int coor_x;
printf("==================================₩n");
printf("Frame buffer Application - ColorBar₩n");
printf("==================================₩n");
if( access(FBDEV_FILE, F_OK) ) // 접근가능한지 체크
{
      printf("%s: access error₩n", FBDEV_FILE);
      exit(1);
}
if( (fb_fd = open(FBDEV_FILE, O_RDWR)) < 0) // frame buffer 드라이버 open
{
      printf("%s: open error₩n", FBDEV_FILE);
      exit(1);
}
if( ioctl(fb_fd, FBIOGET_VSCREENINFO, &fbvar) ) // graphic에 대한 정보 획득
{
      printf("%s: ioctl error - FBIOGET_VSCREENINFO ₩n", FBDEV_FILE);
      exit(1);
}
if( ioctl(fb_fd, FBIOGET_FSCREENINFO, &fbfix) )
{
      printf("%s: ioctl error - FBIOGET_FSCREENINFO ₩n", FBDEV_FILE);
      exit(1);
}
screen_width      =     fbvar.xres;
screen_height     =     fbvar.yres;
bits_per_pixel    =     fbvar.bits_per_pixel;
line_length       =     fbfix.line_length;

printf("screen_width : %d₩n", screen_width);
```

```
    printf("screen_height : %d₩n", screen_height);
    printf("bits_per_pixel : %d₩n", bits_per_pixel);
    printf("line_length : %d₩n", line_length);
// screen의 size 만큼 frame buffer에 메모리 할당
    mem_size      =     screen_width * screen_height * 4;
    fb_mapped     =     (unsigned char *)mmap(0, mem_size,
                        PROT_READ|PROT_WRITE, MAP_SHARED, fb_fd, 0);
    if (fb_mapped < 0)
    {
        printf("mmap error!₩n");
        exit(1);
    }
    for(coor_y = 0; coor_y < screen_height; coor_y++) // 화면을 clear
    {
        ptr =     (unsigned long *)fb_mapped + screen_width * coor_y;
        for(coor_x = 0; coor_x < screen_width; coor_x++)
        {
            *ptr++    =     0x000000;
        }
    }
    int offsety1 = screen_height * 3 / 5; // 화면의 상하 분할점 1
    int offsety2 = screen_height * 4 / 5; // 화면의 상하 분할점 2
    // color bar
    int offsetx1 = screen_width /7 ; // 제일 위쪽의 bar는 가로로 7개
    for(coor_y = 0; coor_y < offsety1; coor_y++)
    {
        ptr =     (unsigned long*)fb_mapped + screen_width * coor_y;
        for (coor_x = 0; coor_x < offsetx1; coor_x++)
        {
            *ptr++    =     0xFFFFFF;
        }
        for (coor_x = offsetx1; coor_x < offsetx1*2; coor_x++)
        {
            *ptr++    =     0xFFE146;
        }
        for (coor_x = (offsetx1*2); coor_x < offsetx1*3; coor_x++)
```

```
        {
            *ptr++   =    0x00D7FF;
        }
        for (coor_x = (offsetx1*3); coor_x < offsetx1*4; coor_x++)
        {
            *ptr++   =    0x006400;
        }
        for (coor_x = (offsetx1*4); coor_x < offsetx1*5; coor_x++)
        {
            *ptr++   =    0xE65AE6;
        }
        for (coor_x = (offsetx1*5); coor_x < offsetx1*6; coor_x++)
        {
            *ptr++   =    0xFF0000;
        }
        for (coor_x = (offsetx1*6); coor_x < screen_width; coor_x++)
        {
            *ptr++   =    0x0000FF;
        }
    }
    for(coor_y = offsety1; coor_y < offsety2; coor_y++)
    {
        ptr =    (unsigned long*)fb_mapped + screen_width * coor_y;
        for (coor_x = 0; coor_x < offsetx1; coor_x++)
        {
            *ptr++   =    0x0000FF;
        }
        for (coor_x = offsetx1; coor_x < offsetx1*2; coor_x++)
        {
            *ptr++   =    0x000000;
        }
        for (coor_x = (offsetx1*2); coor_x < offsetx1*3; coor_x++)
        {
            *ptr++   =    0xE65AE6;
        }
        for (coor_x = (offsetx1*3); coor_x < offsetx1*4; coor_x++)
```

```
        {
            *ptr++   =    0x000000;
        }
        for (coor_x = (offsetx1*4); coor_x < offsetx1*5; coor_x++)
        {
            *ptr++   =    0x00D7FF;
        }
        for (coor_x = (offsetx1*5); coor_x < offsetx1*6; coor_x++)
        {
            *ptr++   =    0x000000;
        }
        for (coor_x = (offsetx1*6); coor_x < screen_width; coor_x++)
        {
            *ptr++   =    0xFFFFFF;
        }
    }
    offsetx1 = screen_width/4; // 가장 낮은 곳의 bar는 4개
    for(coor_y = offsety2; coor_y < screen_height ; coor_y++)
    {
        ptr =    (unsigned long*)fb_mapped + screen_width * coor_y;
        for (coor_x = 0; coor_x < offsetx1; coor_x++)
        {
            *ptr++   =    0xA0A0A0;
        }
        for (coor_x = offsetx1; coor_x < offsetx1*2; coor_x++)
        {
            *ptr++   =    0xFFFFFF;
        }
        for (coor_x = offsetx1*2; coor_x < offsetx1*3; coor_x++)
        {
            *ptr++   =    0x14148C;
        }
        for (coor_x = offsetx1*3; coor_x < screen_width; coor_x++)
        {
            *ptr++   =    0x000000;
        }
```

```
    }
    munmap( fb_mapped, mem_size); // 메모리 해제
    close( fb_fd); // 드라이버 닫음.
    return 0;
}
```

② 컴파일을 수행 후 실행 파일을 타겟 장비에 전송한다.

```
cndi@cndi-virtual:~$ cd colorbar/
cndi@cndi-virtual:~/colorbar$ ls
colorbar   colorbar.c
cndi@cndi-virtual:~/colorbar$ ls
colorbar.c
cndi@cndi-virtual:~/colorbar$ arm-linux-gnueabihf-gcc colorbar.c -o colorbar
cndi@cndi-virtual:~/colorbar$ ls
colorbar   colorbar.c
cndi@cndi-virtual:~/colorbar$scp colorbar ecube@192.168.10.199:/home/ecube/
ecube@192.168.10.199's password:
colorbar                      100%    13KB   12.5KB/s    00:00
cndi@cndi-virtual:~/colorbar$
```

③ 타겟 장비에서 실행 colorbar 실행

```
ecube@udoo:~$ ./colorbar
====================================
Frame buffer Application - ColorBar
====================================
screen_width : 1024
screen_height : 600
bits_per_pixel : 32
line_length : 4096
ecube@udoo:~$
```

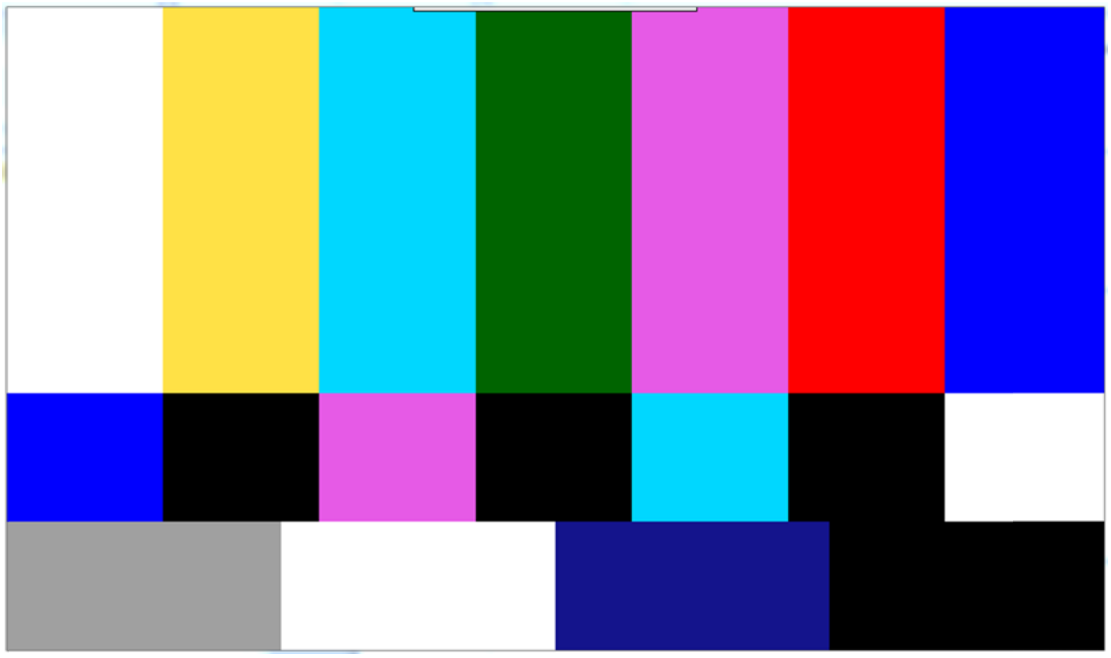

그림 1-1 colorbar 실행 LCD 화면