디바이스드라이버 프로그래밍(2)

ioctl() 시스템 호출

- ioctl()
  - read(), write() 이외의 장치에 특화된 입출력, 제어 동작을 수행하고자 할 때 사용되는 시스템 호출

```
#include <sys/ioctl.h>
int ioctl(int fd, int cmd, ...)
```

  - cmd 인수로 수행할 동작 지정
  - 추가적인 인수 사용 가능

ioctl() 시스템 호출의 구현

- old version에서는 디바이스 드라이버에서 다음 함수 제공
  - linux kernel 2.6.36이후에는 제거됨
  - big kernel lock (BKL) 상태에서 수행되어 성능에 문제 유발

```
int (*ioctl) (struct inode *inode, struct file *filp, unsigned int cmd,
              unsigned long arg);
```

- new version에서는 다음 함수로 변경됨
  - linux kernel 2.6.11부터 추가됨

```
long (*unlocked_ioctl) (struct file *, unsigned int, unsigned long);
long (*compat_ioctl) (struct file *, unsigned int, unsigned long);
```


HEX LED 디바이스 드라이버

- HEX LED
  - DE1-SoC는 6개의 HEX LED 제공
  - 주소 (32-bit I/O)
    - HEX3-HEX0 : 0xFF200020
    - HEX4-HEX5 : 0xFF200030
- 디바이스 드라이버의 예
  - 6개의 HEX LED를 1개의 device로 다룸 (/dev/hex)
  - write() 함수 – 정수를 16진수로 출력 (24비트만 출력됨)
  - read() 함수 – 현재 출력된 정수 값을 읽음
  - **ioctl() 함수** – 출력 형태 제어
    - leading zero를 출력 (기본)
    - leading zero를 출력하지 않음 - cmd의 bit 2 = 1
    - 출력이 깜박임(blink) - cmd의 bit 3 = 1

디바이스 드라이버 구현

■ hex.c

```c
#include <linux/kernel.h>
#include <linux/module.h>
#include <linux/init.h>
#include <linux/interrupt.h>
#include <asm/io.h>

#include <linux/fs.h>
#include <linux/uaccess.h>
#include <linux/types.h>
#include <linux/ioport.h>

MODULE_LICENSE("GPL");
MODULE_AUTHOR("SangKyun Yun");
MODULE_DESCRIPTION("Seven Segment LEDs");

#define base_lwFPGA     0xFF200000
#define len_lwFPGA      0x200000

#define addr_LED        0
#define addr_HEX0       0x20
#define addr_HEX1       0x30
#define addr_SW         0x40
#define addr_KEY        0x50

#define HEX_DEVMAJOR    240
#define HEX_DEVNAME     "hex"

static void  *mem_base;
static void  *hex0_addr;        // HEX3-HEX0
static void  *hex1_addr;        // HEX5-HEX4
static unsigned int data = -1;

static unsigned int mode = 0;
#define NOFILL  4       // bit 2
#define BLINK   8       // bit 3

unsigned int hex0, hex1;        // HEX LED output data
static int turnoff = 0;         // for blink mode
```


write

■ write 함수

```c
int hex_conversion[16] = {
        0x3F, 0x06, 0x5B, 0x4F, 0x66, 0x6D, 0x7D, 0x07,
        0x7F, 0x67, 0x77, 0x7C, 0x39, 0x5E, 0x79, 0x71,
};

static ssize_t hex_write (struct file *file, const char __user *buf, size_t count,
loff_t *f_pos){
        unsigned int hex_data = 0;
        unsigned int nofill = 0;

        get_user(hex_data, (unsigned int *)buf);
//      copy_from_user(&hex_data, buf, count);
        hex_data = hex_data & 0xFFFFFF;         // 24-bit mask
        data = hex_data;                        // for read

        if (mode & NOFILL) nofill = 1;

        hex1 = 0;
        hex0 = hex_conversion[hex_data & 0xf];
```

HEX LED off 상태

write(계속)

```c
        do {
                hex_data >>= 4;
                if (nofill && hex_data==0) break;
                hex0 |= hex_conversion[hex_data & 0xf]<<8;

                hex_data >>= 4;
                if (nofill && hex_data==0) break;
                hex0 |= hex_conversion[hex_data & 0xf]<<16;

                hex_data >>= 4;
                if (nofill && hex_data==0) break;
                hex0 |= hex_conversion[hex_data & 0xf]<<24;

                hex_data >>= 4;
                if (nofill && hex_data==0) break;
                hex1 = hex_conversion[hex_data & 0xf];

                hex_data >>= 4;
                if (nofill && hex_data==0) break;
                hex1 |= hex_conversion[hex_data & 0xf]<<8;
        } while (0);

        iowrite32(hex0, hex0_addr);
        iowrite32(hex1, hex1_addr);

        return 4;
}
```

no zero fill이고
남은 data가 0이면
break

break를 사용하기 위해서
1번 반복하는 do-while 문을 사용

open, release, read, ioctl

```c
static int hex_open(struct inode *minode, struct file *mfile)
{
        return 0;
}

static int hex_release(struct inode *minode, struct file *mfile)
{
        return 0;
}

static ssize_t hex_read (struct file *file, char __user *buf, size_t count,
loff_t *f_pos){
        put_user(data, (unsigned int *)buf);
        return 4;
}

static long hex_ioctl(struct file *file, unsigned int cmd, unsigned long arg)
{
        unsigned int newcmd;

        mode = cmd;
        return 0;
}
```

- file operations

```
static struct file_operations hex_fops = {
        .read           =       hex_read,
        .write          =       hex_write,
        .open           =       hex_open,
        .release        =       hex_release,
//      .ioctl          =       hex_ioctl,
        .unlocked_ioctl =       hex_ioctl,
};
```

- module exit

```
static void __exit hex_exit(void){
        unregister_chrdev(HEX_DEVMAJOR, HEX_DEVNAME);
        printk(" %s unregisterd.\n",HEX_DEVNAME);

        iowrite32(0, hex0_addr);        // turn off all HEX LEDs
        iowrite32(0, hex1_addr);

        iounmap(mem_base);
}

module_init(hex_init);
module_exit(hex_exit);
```


- module init

```
static int __init hex_init(void)
{
        int res;

        res=register_chrdev(HEX_DEVMAJOR, HEX_DEVNAME, &hex_fops);
        if(res < 0) {
                printk(KERN_ERR "hex : failed to register device.\n");
                return res;
        }

        mem_base = ioremap_nocache(base_lwFPGA, len_lwFPGA);
        if( !mem_base) {
                printk("Error mapping memory\n");
                release_mem_region(base_lwFPGA, len_lwFPGA);
                return -EBUSY;
        }
        printk("Device: %s MAJOR: %d\n", HEX_DEVNAME, HEX_DEVMAJOR);

        hex0_addr = mem_base + addr_HEX0;
        hex1_addr = mem_base + addr_HEX1;

        return 0;
}
```


응용 프로그램

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

#include <sys/types.h>
#include <sys/ioctl.h>
#include <sys/stat.h>

#define NOFILL  4
#define BLINK   8

int main(void)
{
        int dev, data, rdata;

        dev = open("/dev/hex", O_RDWR);
        if (dev < 0) {
                fprintf(stderr, "cannot open LED device\n");
                return 1;
        }
// ioctl - default
        ioctl(dev, 0, NULL);
// write
        printf("Input HEX7 data (hex) : ");
        scanf("%x", &data);
        write(dev, &data, 4);
// read
        read(dev, &rdata, 4);
        printf("read data = %x\n", rdata);
// write - NOFILL
        printf("Input HEX7 data (hex) for NOFILL : ");
        scanf("%x", &data);
        ioctl(dev, NOFILL, NULL);
        write(dev, &data, 4);
}
```


- 컴파일

  # make … 커널모듈

  # cc app_hex.c –o app_hex … 응용프로그램

- HEX LED 장치 파일 생성

  # mknod /dev/hex c 240 0 … 생성

  # ls -l /dev/hex … 확인

  crw-r--r-- 1 root root 240, 0 Jan 1 00:29 /dev/hex

- 장치드라이버 설치 및 제거

  # insmod hex.ko … 설치

  # lsmod … 확인

  Module Size Used by

  hex 1694 0

  # 응용프로그램 실행

  …

  # rmmod hex … 제거

  # lsmod … 확인

타이머 사용

- timer 사용
  - init_timer() – timer 초기화
  - add_timer() – timer 함수 등록
  - del_timer() – timer함수 제거

  - **void init_timer(struct timer_list *timer);**
  - **void add_timer(struct timer_list *timer);**
  - **int del_timer(struct timer_list *timer);**

```
struct timer_list {
    struct timer_list *next;          /* never touch this */
    struct timer_list *prev;          /* never touch this */
    unsigned long expires;            /* the timeout, in jiffies */
    unsigned long data;               /* argument to the handler */
    void (*function)(unsigned long);  /* handler of the timeout */
    volatile int running;             /* added in 2.4; don't touch */
};
```



```
struct timer_list hex_timer;

void init_add_timer(void)
{
        init_timer(&hex_timer);

        hex_timer.function = hex_timer_function;
        hex_timer.expires = jiffies + HZ;       // after 1 sec
        hex_timer.data = 0;

        add_timer(&hex_timer);
}

void remove_timer(void)
{
        del_timer(&hex_timer);
}

void hex_timer_function(unsigned long ptr)
{
        if ( !(mode & BLINK) ) return;
        turnoff = !turnoff;
        if (turnoff) {
                iowrite32(0, hex0_addr);
                iowrite32(0, hex1_addr);
        } else {
                iowrite32(hex0, hex0_addr);
                iowrite32(hex1, hex1_addr);
        }
        init_add_timer();
}
```


ioctl – BLINK 포함

```
static long hex_ioctl(struct file *file, unsigned int cmd, unsigned long arg)
{
        unsigned int newcmd;

        newcmd = cmd;
        if ( (mode & BLINK) && !(newcmd & BLINK))
            remove_timer();
        mode = newcmd;
        if (mode & BLINK) {
                init_add_timer();
        }
        return 0;
}
```