[실습 10] 디바이스드라이버프로그래밍(2)

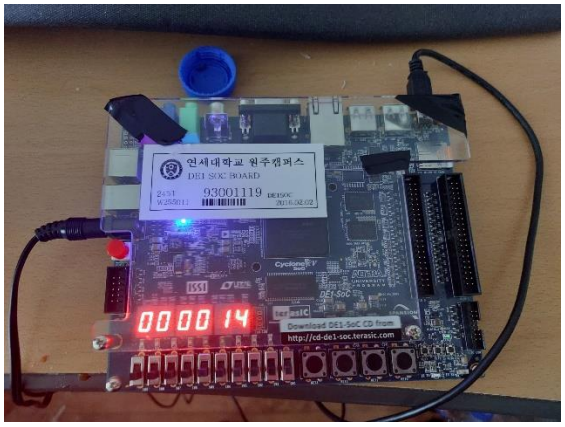
2017253019 안희영

1. (7 세그먼트 HEX LED 디바이스 드라이버)

(1) 주어진 소스파일 내용을 참고하여 7 세그면트 HEX 디바이스 드라이버를 디바이스를 자동으로 생성하는 방법을 사용하고, ioctl()루틴을 NOFILL 과 BLINKING 지원하도록 timer interrupt 등을 활용하여 작성하고, 컴파일 후에 커널모듈을 설치하시오. 설치 전후에 디바이스가 존재하는 지 여부를 각각 확인하시오.

```
root@de1soclinux:~/hw4# make
make -C /lib/modules/3.13.0-00293-g1664d72/build M=/home/root/hw4 modules
make[1]: Entering directory `/usr/src/3.13.0-00293-g1664d72'
  Building modules, stage 2.
  MODPOST 1 modules
make[1]: Leaving directory `/usr/src/3.13.0-00293-g1664d72'
root@de1soclinux:~/hw4# ls ./dev
ls: cannot access ./dev: No such file or directory
root@de1soclinux:~/hw4# ls /.dev
ls: cannot access /.dev: No such file or directory
root@de1soclinux:~/hw4# ls /dev
block             network_throughput  ram0       tty20  tty40  tty60  ttypd
bus               null                ram1       tty21  tty41  tty61  ttype
char              psaux               random     tty22  tty42  tty62  ttypf
console           ptmx                shm        tty23  tty43  tty63  urandom
cpu_dma_latency   ptp0                spidev0.0  tty24  tty44  tty7   vcs
disk              pts                 stderr     tty25  tty45  tty8   vcs1
fd                ptyp0               stdin      tty26  tty46  tty9   vcs2
fpga0             ptyp1               stdout     tty27  tty47  ttyS0  vcs3
full              ptyp2               tty        tty28  tty48  ttyS1  vcs4
gator             ptyp3               tty0       tty29  tty49  ttyp0  vcs5
i2c-0             ptyp4               tty1       tty3   tty5   ttyp1  vcs6
i2c-1             ptyp5               tty10      tty30  tty50  ttyp2  vcs7
input             ptyp6               tty11      tty31  tty51  ttyp3  vcsa
kmem              ptyp7               tty12      tty32  tty52  ttyp4  vcsa1
kmsg              ptyp8               tty13      tty33  tty53  ttyp5  vcsa2
log               ptyp9               tty14      tty34  tty54  ttyp6  vcsa3
mem               ptypa               tty15      tty35  tty55  ttyp7  vcsa4
mmcblk0           ptypb               tty16      tty36  tty56  ttyp8  vcsa5
mmcblk0p1         ptypc               tty17      tty37  tty57  ttyp9  vcsa6
mmcblk0p2         ptypd               tty18      tty38  tty58  ttypa  vcsa7
mmcblk0p3         ptype               tty19      tty39  tty59  ttypb  watchdog
network_latency   ptypf               tty2       tty4   tty6   ttypc  zero
root@de1soclinux:~/hw4# ls
Makefile        app_hex     hex_cl.c   hex_cl.mod.c  hex_cl.o
Module.symvers  app_hex.c   hex_cl.ko  hex_cl.mod.o  modules.order
root@de1soclinux:~/hw4# insmod hex_cl.ko
root@de1soclinux:~/hw4# lsmod
Module                  Size  Used by
hex_cl                  2764  0
root@de1soclinux:~/hw4# ls /dev
block             network_throughput  ram1       tty22  tty43  tty7     vcs1
bus               null                random     tty23  tty44  tty8     vcs2
char              psaux               shm        tty24  tty45  tty9     vcs3
console           ptmx                spidev0.0  tty25  tty46  ttyS0    vcs4
cpu_dma_latency   ptp0                stderr     tty26  tty47  ttyS1    vcs5
disk              pts                 stdin      tty27  tty48  ttyp0    vcs6
fd                ptyp0               stdout     tty28  tty49  ttyp1    vcs7
fpga0             ptyp1               tty        tty29  tty5   ttyp2    vcsa
full              ptyp2               tty0       tty3   tty50  ttyp3    vcsa1
gator             ptyp3               tty1       tty30  tty51  ttyp4    vcsa2
hex               ptyp4               tty10      tty31  tty52  ttyp5    vcsa3
i2c-0             ptyp5               tty11      tty32  tty53  ttyp6    vcsa4
i2c-1             ptyp6               tty12      tty33  tty54  ttyp7    vcsa5
input             ptyp7               tty13      tty34  tty55  ttyp8    vcsa6
kmem              ptyp8               tty14      tty35  tty56  ttyp9    vcsa7
kmsg              ptyp9               tty15      tty36  tty57  ttypa    watchdog
log               ptypa               tty16      tty37  tty58  ttypb    zero
mem               ptypb               tty17      tty38  tty59  ttypc
mmcblk0           ptypc               tty18      tty39  tty6   ttypd
mmcblk0p1         ptypd               tty19      tty4   tty60  ttype
mmcblk0p2         ptype               tty2       tty40  tty61  ttypf
mmcblk0p3         ptypf               tty20      tty41  tty62  urandom
network_latency   ram0                tty21      tty42  tty63  vcs
root@de1soclinux:~/hw4# ./app_hex
Input HEX7 data (hex) : 14
read data = 14
Input HEX7 data (hex) for BLINK : 11
Input HEX7 data (hex) for NOFILL : 10
Input HEX7 data (hex) for NOFILL, BLINK : 9
root@de1soclinux:~/hw4#
```

모듈을 설치한 이후 hex 디바이스가 추가된 것을 확인할 수 있습니다.



기본적인 출력상태입니다.

blinking 으로 인해 꺼진 상태입니다.

nofill 이 적용되어 있습니다.

위의 터미널 이미지의 입력대로 출력되는걸 확인할 수 있습니다.

(2) 함께 주어진 작성한 HEX LED 용 디바이스 드라이버를 테스트하는 프로그램 app_hex.c 를 분석하고, 이를 작성한 후 동작을 확인하시오.

```
printf("Input HEX7 data (hex) for NOFILL : ");
scanf("%x", &data);
ioctl(dev, NOFILL, NULL);
write(dev, &data, 4);
```

출력할 데이터를 입력받은후 ioctl 로 device 의 모드와 컨트롤할 디바이스의 번호를 전달합니다..(디바이스의 번호가 동적으로 할당되기 때문입니다.)

Write 함수에 device 의 번호와 데이터, 그리고 크기를 전달합니다.

(3) 디바이스를 자동으로 생성하려면 init 함수에서 어떤 과정을 거치는가? 그리고 exit 함수에서는 어떤 동작을 수행하는가? (참고: blink 모드를 지원할 때에는 현재 blink 모드가 설정되어 있으면 exit 함수에서 timer 도 제거해야 한다.)

alloc_chrdev_region(&dev_no,0,1,DEVICE_NAME) 함수로 디바이스 번호를 할당받습니다.

cdev_init(&hex_cdev,&hex_fops) 로 cdev 구조체를 초기화한 후 fops 를 지정합니다.

cdev_add(&hex_cdev,dev_no,1) 구조체를 등록합니다.

class_create (THIS_MODULE, DEVICE_NAME) device_create 호출 시 사용할 class 구조체 생성, 포인터를 반환합니다.

device_create (cl,NULL,dev_no,NULL,DEVICE_NAME) device 를 생성하고, sysfs 에 등록합니다.

Remove_timer() timer 를 제거합니다. – del_timer(&hex_timer) 함수를 실행하는 합수입니다.

device_destroy(cl,dev_no) 생성된 device 를 제거합니다.

class_destroy(cl) class_create 로 생성한 class 구조체를 제거합니다.

unregister_chdev_region(dev_no,1) 할당된 디바이스 번호를 해제합니다.

(4) write 함수에서 nofill 을 어떻게 구현하였는지 설명하시오.

Nofill 모드인 경우 nofill 을 1 로 세트한다.

Hex_data 를 4 비트씩 shift 연산후 4 비트를 마스킹하여 값이 존재하지 않으면 hex 출력값에 값을 더하는걸 멈추고 루프를 빠져나온다.

(5) blink 출력을 어떻게 구현하였는지 설명하시오.

```
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();
}
```

일정 시간에 따라 turnoff 상태를 바꾸고 아무런 출력값이 없는 hex 값과 출력값이 들어있는 hex 값을 차례대로 hexled 의 출력으로 내보냅니다. 그리고 다음 상태변화를 위한 인터럽트를 발생시키기 위해 timer 를 다시 세트합니다.

해당 루틴을 hex_timer.function 에 설정하여 timeintterupt 가 발생한 경우 실행되게 합니다.