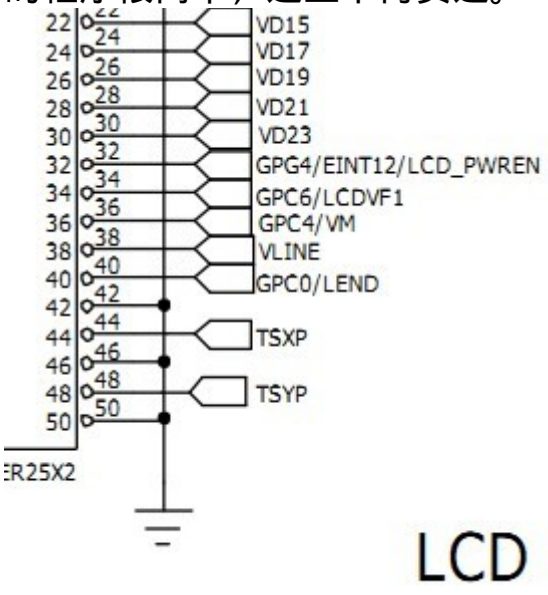
努力成为 linux kernel hacker 的人李万鹏原创作品，为梦而战。转载请标明出处
http://blog.csdn.net/woshixingaaa/archive/2011/05/16/6423462.aspx
下图是 LCD 的引脚截图，GPG4 是复用引脚，可以发送电源使能信号。GPIO 的程序很简单，这里不再赘述。



LCD

驱动程序：

```
1. #include <linux/init.h>
2. #include <linux/module.h>
3. #include <linux/fs.h>
4. #include <mach/regs-gpio.h>
5. #include <asm/io.h>
6. #include <mach/hardware.h>
7. #include <linux/device.h>
8. #include <linux/cdev.h>
9. #include <linux/delay.h>
10.int MYGPIO_MAJOR = 0;
11.int MYGPIO_MINOR = 0;
12.#define MYGPIO_NAME "lwp-gpio"
13.dev_t dev_num;
14.struct cdev *gpio_cdev;
15.struct class *gpio_class;
16.int i;
17.static unsigned long led_table[]={
18.    S3C2410_GPF3,
19.    S3C2410_GPF4,
20.    S3C2410_GPF5,
21.    S3C2410_GPF6,
22.};
23.static unsigned long led_cfg_table[]={
24.    S3C2410_GPF3_OUTP,
```

```
    S3C2410_GPF4_OUTP,
    S3C2410_GPF5_OUTP,
    S3C2410_GPF6_OUTP,
};
int gpio_ioctl(struct inode *inode, struct file *file, unsigned int cmd, unsigned long arg){
    switch(cmd){
        case -1:
            s3c2410_gpio_cfgpin(S3C2410_GPG4, S3C2410_GPG4_OUTP);        //关闭 LCD 背光
            s3c2410_gpio_setpin(S3C2410_GPG4, 0);
            break;
        case  1:                                                    //打开背光
            s3c2410_gpio_setpin(S3C2410_GPG4, 1);
            break;
        case  2:                                                    //开启流水灯
            while(1){
                for(i = 0; i < 4; i++)
                    s3c2410_gpio_cfgpin(led_table[i], led_cfg_table[i]);
                for(i = 0; i < 4; i++){
                    s3c2410_gpio_setpin(led_table[i],0);
                    msleep(1000);
                    s3c2410_gpio_setpin(led_table[i],1);
                }
            }
            break;
        case -2:                                                    //关闭流水灯
            for(i = 0; i < 4; i++){
                s3c2410_gpio_setpin(led_table[i],1);
            }
            break;
        case  3:                                                    //开启蜂鸣器
            s3c2410_gpio_cfgpin(S3C2410_GPB0, S3C2410_GPB0_OUTP);
            s3c2410_gpio_setpin(S3C2410_GPB0, 1);
            break;
        case -3:                                                    //关闭蜂鸣器
            s3c2410_gpio_setpin(S3C2410_GPB0, 0);
            break;
        default:
            break;
    }
    return 0;
}
int gpio_open(struct inode *inode, struct file *file){
    printk("gpio is opened success/n");
    return 0;
}
int gpio_close(struct inode *inode, struct file *file){
```

```
    printk("gpio is closed success/n");
    return 0;
}
struct file_operations gpio_ops = {
    .owner = THIS_MODULE,
        .ioctl = gpio_ioctl,
    .open = gpio_open,
    .release = gpio_close,
};
static int __init my_gpio_init(void){
    int ret;
    ret = alloc_chrdev_region(&dev_num, MYGPIO_MINOR, 1, MYGPIO_NAME);          //分配设备号
    if(ret < 0)
        printk("can't get major number/n");
    gpio_cdev = kmalloc(sizeof(struct cdev),GFP_KERNEL);                        //分配字父设备
    if(!gpio_cdev){
        return -ENOMEM;
        goto fail_malloc;
    }
    memset(gpio_cdev,0,sizeof(struct cdev));
    cdev_init(gpio_cdev, &gpio_ops);                                            //初始化字父设备
    cdev_add(gpio_cdev, dev_num, 1);                                            //字符设备注册到系统
    gpio_class = class_create(THIS_MODULE,MYGPIO_NAME);                         //在/sys 下建立一个类
    if(IS_ERR(gpio_class)){
        printk("ERROR: Fail to create gpio_class class/n");
        return -1;
    }
    device_create(gpio_class,NULL,dev_num,NULL,MYGPIO_NAME);                    //创建设备节点
    printk("gpio_cdev is registered success/n");
fail_malloc:
    unregister_chrdev_region(dev_num, 1);
    return 0;
}
static void __exit my_gpio_exit(void){
    unregister_chrdev_region(dev_num, 1);
    cdev_del(gpio_cdev);
    kfree(gpio_cdev);
    device_destroy(gpio_class,dev_num);
    class_destroy(gpio_class);
    printk("gpio_cdev is dereigstered success/n");
}
module_init(my_gpio_init);
module_exit(my_gpio_exit);
MODULE_LICENSE("GPL");
MODULE_AUTHOR("liwanpeng");
```

测试程序：

```c
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <sys/ioctl.h>
int main(){
    int fd, cmd;
    cmd = 0;
    fd = open("/dev/lwp-gpio",O_RDWR);
    if(fd < 0){
        printf("cannot open /dev/lwp-gpio/n");
        exit(1);
    }
    while(1){
        scanf("%d", &cmd);
        printf("cmd is %d/n",cmd);
        ioctl(fd, cmd);
    }
    close(fd);
    return 0;
}
```