

磁盘IO

林仲航 无63 2016011051

1、问题描述：

通过针对磁盘进行I/O实验，了解与掌握直接访问磁盘扇区的方法。要求实现三个函数：

1. physicalDisk：判定逻辑驱动器X中磁盘的基本信息
2. sectorRead：根据给定的物理扇区号读取磁盘的扇区
3. sectorDump：查看磁盘的内容并把磁盘上得到的信息输出到标准输出流。

2、实验环境：

Ubuntu16.04LTS

3、函数实现：

(1) sectorRead:

该函数较简单实现，通过open函数打开对应磁盘，设要读取第x个扇区，通过ioctl(fd,BLKSSZGET,&size) 可以获取一个扇区的大小size，进而可以利用read函数进行读取，详细的函数如下：

```
int get_sector_size(int fd){
    //get sector size of fd
    int size;
    ioctl(fd,BLKSSZGET,&size);
    return size;
}

int sectorRead(int fd, unsigned long sectorID, char **p){
    int sector_size = get_sector_size(fd);
    lseek(fd,0,SEEK_SET);           //指针移动到开头
    if(lseek(fd,sectorID * sector_size,SEEK_CUR) == -1){
        //将指针移动到对应位置
        std::cerr << "no such sector" << std::endl;
    }
    //读取一扇区的数据
    *p = new char[sector_size];
    return read(fd,*p,sector_size);
}
```

(2) sectorDump:

该函数实现将读取的信息通过16进制显示出来。我们只需要将sectorRead中读取的数据转换为16进制数即可。详细函数如下：

```
void sectorDump(char *p, int size){
    int line_num = 16;           //每行显示的数据数
    int i = 0;
    while( i < size ){
        unsigned char a = (*(p+i)); //读取第i个数据(char)
        if( line_num == 0 ){
            std::cout << std::endl;
            line_num = 16;
        }
        else{
            unsigned short l = a & 0x0f;
            unsigned short h = (a & 0xf0)>>4;
            std::cout.setf(std::ios_base::right, std::ios_base::adjustfield);
            std::cout.fill(' ');
            std::cout.width(2);
            std::cout <<std::hex<< h << l ;
            //将数据按16进制输出

            line_num --;
            if(line_num % 4 ==0 )
                std::cout << " ";
        }
        i++;
    }
    std::cout << std::endl;
}
```

(3) physicalDisk:

此函数读取磁盘驱动器的基本信息。在linux下读取设备的基本信息接口为ioctl。

首先考虑读取磁盘结构，根据源码查到如下结构体：

```
struct hd_geometry {
    unsigned char heads;
    unsigned char sectors;
    unsigned short cylinders;
    unsigned long start;
};
```

此结构描述磁盘布局，故而我们可以通过如下代码得到磁盘布局并输出：

```
hd_geometry hdio;
ioctl(fd, HDIO_GETGEO, &hdio);
```

此外可以通过标识符BLKGETSIZE64得到磁盘大小信息：

```
long long disk_size;
ioctl(fd, BLKGETSIZE64, &disk_size);
```

获取扇区大小通过之前定义的函数get_sector_size即可。

综上，physicalDisk代码如下：

```
void physicalDisk(int fd){
    //get basic infomation about disk
    int size;
    ioctl(fd, BLKSSZGET, &size);
    //获取扇区大小
    long long disk_size;
    ioctl(fd, BLKGETSIZE64, &disk_size);
    //获取磁盘大小
    hd_geometry hdio;
    ioctl(fd, HDIO_GETGEO, &hdio);
    //获取磁盘布局

    //输出磁盘信息
    std::cout << "sector size: " << size << " B " << std::endl
        << "disk size: " << disk_size/size << std::endl
        << "磁头:" << int(hdio.heads) << std::endl
        << "柱面:" << hdio.cylinders << std::endl
        << "扇区:" << int(hdio.sectors) << std::endl
        << "起始:" << hdio.start << std::endl;
}
```

(4)、测试程序：

我们通过对磁盘/dev/sda进行读取基本信息以及读取第0个扇区并输出。程序如下：

```
int main(int argc, char *argv[])
{
    int fd = open("/dev/sda", O_RDONLY);
    std::cout << fd << std::endl;

    physicalDisk(fd);
    char * p;

    int a = sectorRead(fd, 0, &p);
    if(a>=0){
        std::cout << "read success " << a << " bytes " << std::endl;
        sectorDump(p, a);
    }
}
```

```

    }
    else{
        std::cout << "reading error" << std::endl;
    }
}

```

4、验证：

在root下编译main.cpp并运行：

```

g++ main.cpp -o main
./main

```

得到如下结果：

```

root@ubuntu: /home/robin/EE_Operate_System_Term_Project/file_system/src
main
root@ubuntu: /home/robin/EE_Operate_System_Term_Project/file_system/src# ./main
3
sector size: 512 B
sectors: 146800640
磁头:255
柱面:9137
扇区:63
起始:0
read success 512 bytes
eb 63 90 10 8e d0 bc 00 b0 b8 00 00 8e d8 8e c0
be 00 7c bf 00 06 b9 00 02 f3 a4 ea 21 06 00 00
be 07 38 04 75 0b 83 c6 10 81 fe fe 07 75 f3 eb
b4 02 b0 01 bb 00 7c b2 80 8a 74 01 8b 4c 02 cd
ea 00 7c 00 00 eb fe 00 00 00 00 00 00 00 00
00 00 00 00 00 00 80 01 00 00 00 00 00 00 ff
90 90 f6 c2 80 74 05 f6 c2 70 74 02 b2 80 ea 79
00 00 31 c0 8e d8 8e d0 bc 00 20 fb a0 64 7c 3c
74 02 88 c2 52 bb 17 04 f6 07 03 74 06 be 88 7d
17 01 be 05 7c b4 41 bb aa 55 cd 13 5a 52 72 3d
fb 55 aa 75 37 83 e1 01 74 32 31 c0 89 44 04 40
44 ff 89 44 02 c7 04 10 00 66 8b 1e 5c 7c 66 89
08 66 8b 1e 60 7c 66 89 5c 0c c7 44 06 00 70 b4
cd 13 72 05 bb 00 70 eb 76 b4 08 cd 13 73 0d 5a
d2 0f 83 d0 00 be 93 7d e9 82 00 66 0f b6 c6 88
ff 40 66 89 44 04 0f b6 d1 c1 e2 02 88 e8 88 f4
89 44 08 0f b6 c2 c0 e8 02 66 89 04 66 a1 60 7c
09 c0 75 4e 66 a1 5c 7c 66 31 d2 66 f7 34 88 d1
d2 66 f7 74 04 3b 44 08 7d 37 fe c1 88 c5 30 c0
e8 02 08 c1 88 d0 5a 88 c6 bb 00 70 8e c3 31 db
01 02 cd 13 72 1e 8c c3 60 1e b9 00 01 8e db 31
bf 00 80 8e c6 fc f3 a5 1f 61 ff 26 5a 7c be 8e
eb 03 be 9d 7d e8 34 00 be a2 7d e8 2e 00 cd 18
fe 47 52 55 42 20 00 47 65 6f 6d 00 48 61 72 64
44 69 73 6b 00 52 65 61 64 00 20 45 72 72 6f 72
0a 00 bb 01 00 b4 0e cd 10 ac 3c 00 75 f4 c3 28
5b 3b 00 00 80 20 21 00 83 fe ff ff 00 08 00 00
70 a1 08 00 fe ff ff 05 fe ff ff fe 7f a1 08 02
1e 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
55 aa
root@ubuntu: /home/robin/EE_Operate_System_Term_Project/file_system/src#

```

通过命令：

```
hdparm /dev/sda
```

查看磁盘信息：

```
root@ubuntu:/home/robin/EE_Operate_System_Term_Project/file_system/src# hdparm /dev/sda
/dev/sda:
SG_IO: bad/missing sense data, sb[]: 70 00 05 00 00 00 00 0a 00 00 00 00 20 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
multcount      = 0 (off)
readonly       = 0 (off)
readahead      = 256 (on)
geometry       = 9137/255/63, sectors = 146800640, start = 0
```

可以看出结果一致。

通过软件wxhexeditor可以查看磁盘的16进制数据，对/dev/sda的第0个扇区查看得到如下结果：

0.23 Beta for Linux

sda

偏移量	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16
000000000000	EB	63	90	10	8E	D0	BC	00	B0	B8	00	00	8E	D8	8E	C0	FB	BE	00	7C	BF	00	06
000000000023	B9	00	02	F3	A4	EA	21	06	00	00	BE	BE	07	38	04	75	0B	83	C6	10	81	FE	FE
000000000046	07	75	F3	EB	16	B4	02	B0	01	BB	00	7C	B2	80	8A	74	01	8B	4C	02	CD	13	EA
000000000069	00	7C	00	00	EB	FE	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	80
000000000092	01	00	00	00	00	00	00	FF	FA	90	90	F6	C2	80	74	05	F6	C2	70	74	02	B2	00
000000000115	80	EA	79	7C	00	00	31	C0	8E	D8	8E	D0	BC	00	20	FB	A0	64	7C	3C	FF	74	02
000000000138	88	C2	52	BB	17	04	F6	07	03	74	06	BE	88	7D	E8	17	01	BE	05	7C	B4	41	BB
000000000161	AA	55	CD	13	5A	52	72	3D	81	FB	55	AA	75	37	83	E1	01	74	32	31	C0	89	44
000000000184	04	40	88	44	FF	89	44	02	C7	04	10	00	66	8B	1E	5C	7C	66	89	5C	08	66	8B
000000000207	1E	60	7C	66	89	5C	0C	C7	44	06	00	70	B4	42	CD	13	72	05	BB	00	70	EB	76
000000000230	B4	08	CD	13	73	0D	5A	84	D2	0F	83	D0	00	BE	93	7D	E9	82	00	66	0F	B6	C6
000000000253	88	64	FF	40	66	89	44	04	0F	B6	D1	C1	E2	02	88	E8	88	F4	40	89	44	08	0F
000000000276	B6	C2	C0	E8	02	66	89	04	66	A1	60	7C	66	09	C0	75	4E	66	A1	5C	7C	66	31
000000000299	D2	66	F7	34	88	D1	31	D2	66	F7	74	04	3B	44	08	7D	37	FE	C1	88	C5	30	C0
000000000322	C1	E8	02	08	C1	88	D0	5A	88	C6	BB	00	70	8E	C3	31	DB	B8	01	02	CD	13	72
000000000345	1E	8C	C3	60	1E	B9	00	01	8E	DB	31	F6	BF	00	80	8E	C6	FC	F3	A5	1F	61	FF
000000000368	26	5A	7C	BE	8E	7D	EB	03	BE	9D	7D	E8	34	00	BE	A2	7D	E8	2E	00	CD	18	EB
000000000391	FE	47	52	55	42	20	00	47	65	6F	6D	00	48	61	72	64	20	44	69	73	6B	00	52
000000000414	65	61	64	00	20	45	72	72	6F	72	0D	0A	00	BB	01	00	B4	0E	CD	10	AC	3C	00
000000000437	75	F4	C3	28	ED	5B	3B	00	00	80	20	21	00	83	FE	FF	FF	00	08	00	00	00	70
000000000460	A1	08	00	FE	FF	FF	05	FE	FF	FF	FE	7F	A1	08	02	78	1E	00	00	00	00	00	00

光标偏移量：97 光标值：0 选择的区块：不可用 数

经过对比可以看出结果是一样的。函数正确性得到验证。

5、总结：

通过以上的磁盘操作让我对直接磁盘IO有了深入的认识，本来想通过写这些函数来写FAT文件系统作业，但是拖延症的我时间不够，只能将此交上去，甚是遗憾。