VxWorks 的文件大小

在 VxWorks 里如何查看文件的尺寸(size)呢？

在 Shell 下可以使用命令 ll()

```
-> ll "a.txt"
value = 0 = 0x0
-> open "a.txt",2|0x200,0644
value = 6 = 0x6
-> write 6,"hello",5
value = 5 = 0x5
-> close 6
value = 0 = 0x0
-> ll "a.txt"
-rw-rw-rw-  1 0        0                    5 Jan  6 07:56 a.txt
value = 0 = 0x0
-> remove "a.txt"
value = 0 = 0x0
-> ll "a.txt"
value = 0 = 0x0
->
```

而在 code 中，可以通过文件系统的 IO 操作来获取文件的尺寸。毕竟文件也是一种 IO 嘛常用的函数有 lseek()，它的作用本是设置文件读写指针的位置，而它的返回值是从文件开始位置到当前位置的偏移。

```
/* POSIX requires off_t to be a signed int type */
# ifdef __RTP__
typedef long long       off_t;
# else  /* __RTP__ */
typedef long            off_t;

off_t lseek
    (
    int   fd,           /* file descriptor           */
    off_t offset,       /* new byte offset to seek to */
    int   whence
    );
```

参数 whence 取值有三：

- SEEK_SET (0) |- set to <offset>
- SEEK_CUR (1) |- set to current position plus <offset>
- SEEK_END (2) |- set to the size of the file plus <offset>

因此使用 lseek(fd, 0, SEEK_END)就可以得到文件的尺寸了

test_lseek.c

```c
#include <ioLib.h>     /* open() lseek() */
#include <stdIo.h>     /* printf() */

void test()
    {
    int i;
    int fd = open("a.txt",O_RDWR|O_CREAT,0644);   /* 创建文件 */
    (void)write(fd,"hello",5);                    /* 添加5个字符 */

    i = lseek(fd,0,SEEK_END);
    (void)printf("file size = %d\n",i); /* 当前size为5 */

    (void)close(fd);
    (void)remove("a.txt");
    }
```

Target Consoles | Build Console

vxsim0@Yang

```
->
->
-> test
file size = 5
value = 0 = 0x0
```

不过 POSIX 规定了 offset 必须是 int 类型的。如果文件尺寸超过了 32 位数的范围，则可以使用 IO 系统的 FIOFSTATGET 操作

test_stat.c

```c
#include <ioLib.h>     /* open() ... FIOFSTATGET */
#include <stdIo.h>     /* printf() */
#include <sys/stat.h> /* struct stat */

void test()
    {
    struct stat myStat;
    int fd = open("a.txt",O_RDWR|O_CREAT,0644);      /* 创建文件 */
    (void)write(fd,"hello",5);                       /* 添加5个字符 */

    (void)ioctl (fd, FIOFSTATGET, (int)&myStat);
    (void)printf("file size = %d\n",myStat.st_size);/* 文件size为5 */

    (void)close(fd);
    (void)remove("a.txt");
    }
```

Target Consoles | Build Console

vxsim0@Yang

```
->
-> test
file size = 5
value = 0 = 0x0
->
```

成员 st_size 的类型是 long long，足够用了。
POSIX 还定义了两个封装更好的函数 fstat()和 stat()

```
01: STATUS fstat(int fd, struct stat *pStat);
02: STATUS stat(char *name, struct stat *pStat);
```

其实它俩内部就是调用的 FIOFSTATGET 操作

```
1  #include <ioLib.h>     /* open() */
2  #include <stdIo.h>     /* printf() */
3  #include <sys/stat.h> /* struct stat */
4
5  void test()
6      {
7      struct stat myStat;
8      int fd = open("a.txt",O_RDWR|O_CREAT,0644);      /* 创建文件 */
9      (void)write(fd,"hello",5);                      /* 添加5个字符 */
10     (void)close(fd);
11
12     (void)stat ("a.txt", &myStat);
13     (void)printf("file size = %d\n",myStat.st_size);/* 文件size为5 */
14
15     (void)remove("a.txt");
16     }
```

Target Consoles | Build Console
vxsim0@Yang

```
->
-> test
file size = 5
value = 0 = 0x0
->
```

或者借用 IO 系统的 FIONREAD/FIONREAD64 操作来查看一共还有多少字符可以读取，也可以得到文件尺寸

test_read.c

```
1  #include <ioLib.h>     /* open() ... FIOFSTATGET */
2  #include <stdIo.h>     /* printf() */
3
4  void test()
5      {
6      int num;
7      int fd = open("a.txt",O_RDWR|O_CREAT,0644);      /* 创建文件 */
8      (void)write(fd,"hello",5);                      /* 添加5个字符 */
9      (void)lseek(fd,0,SEEK_SET);                      /* 设置文件指针到文件头部 */
10
11     (void)ioctl (fd, FIONREAD, &num);
12     (void)printf("file size = %d\n", num);          /* 文件size为5 */
13
14     (void)close(fd);
15     (void)remove("a.txt");
16     }
```

Target Consoles | Build Console
vxsim0@Yang

```
->
-> test
file size = 5
value = 0 = 0x0
->
```

而要修改文件的尺寸，可以用 FIOTRUNC/FIOTRUNC64

test.c

```
1  #include <ioLib.h>     /* open() ... FIOTRUNC */
2  #include <stdIo.h>     /* printf() */
3  #include <stdLib.h>    /* malloc() free() */
4  #include <string.h>    /* bzero() */
5
6  void test()
7      {
8      int i;
9      char *buf = malloc(10);
10     int fd = open("a.txt",O_RDWR|O_CREAT,0644);   /* 创建文件 */
11     (void)write(fd,"hello",5);                    /* 添加5个字符 */
12
13     i = ioctl(fd, FIOTRUNC, 3);                   /* 裁剪文件size为3 */
14     (void)printf("FIOTRUNC = %d\n",i);            /* 返回值为OK */
15     i = lseek(fd,0,SEEK_END);
16     (void)printf("size after FIOTRUNC = %d\n",i);/* 当前size为3 */
17
18     (void)lseek(fd,0,SEEK_SET);                   /* 将文件指针指向文件头部 */
19     bzero(buf,10);
20     i = read(fd,buf,10);                          /* 读取全部字符 */
21     (void)printf("read %d characters: %s\n", i, buf);/* 仅有3个字符 */
22
23     (void)close(fd);
24     (void)remove("a.txt");
25     free(buf);
26     }
```

Target Consoles | Build Console

vxsim0@Yang

```
->
-> test
FIOTRUNC = 0
size after FIOTRUNC = 3
read 3 characters: hel
value = 0 = 0x0
->
```

ioctl()支持参数还有很多，有兴趣的朋友可以去了解一下,参考下图

```
01: #define FIONREAD        1   /* get num chars available to read */
02: #define FIOFLUSH        2   /* flush any chars in buffers */
03: #define FIOOPTIONS      3   /* set options (FIOSETOPTIONS) */
04: #define FIOBAUDRATE     4   /* set serial baud rate */
05: #define FIODISKFORMAT   5   /* format disk */
06: #define FIODISKINIT     6   /* initialize disk directory */
07: #define FIOSEEK         7   /* set current file char position */
08: #define FIOWHERE        8   /* get current file char position */
09: #define FIODIRENTRY     9   /* return a directory entry (obsolete)*/
10: #define FIORENAME       10  /* rename a directory entry */
11: #define FIOREADYCHANGE  11  /* return TRUE if there has been a media change on
12: #define FIONWRITE       12  /* get num chars still to be written */
13: #define FIODISKCHANGE   13  /* set a media change on the device */
14: #define FIOCANCEL       14  /* cancel read or write on the device */
15: #define FIOSQUEEZE      15  /* OBSOLETED since RT11 is obsoleted */
16: #define FIONBIO         16  /* set non-blocking I/O; SOCKETS ONLY!*/
17: #define FIONMSGS        17  /* return num msgs in pipe */
18: #define FIOGETNAME      18  /* return file name in arg */
19: #define FIOGETOPTIONS   19  /* get options */
20: #define FIOSETOPTIONS   FIOOPTIONS  /* set options */
21: #define FIOISATTY       20  /* is a tty */
22: #define FIOSYNC         21  /* sync to disk */
23: #define FIOPROTOHOOK    22  /* specify protocol hook routine */
24: #define FIOPROTOARG     23  /* specify protocol argument */
25: #define FIORBUFSET      24  /* alter the size of read buffer  */
26: #define FIOWBUFSET      25  /* alter the size of write buffer */
27: #define FIORFLUSH       26  /* flush any chars in read buffers */
28: #define FIOWFLUSH       27  /* flush any chars in write buffers */
29: #define FIOSELECT       28  /* wake up process in select on I/O */
30: #define FIOUNSELECT     29  /* wake up process in select on I/O */
31: #define FIONFREE        30  /* get free byte count on device */
32: #define FIOMKDIR        31  /* create a directory */
33: #define FIORMDIR        32  /* remove a directory */
34: #define FIOLABELGET     33  /* get volume label */
35: #define FIOLABELSET     34  /* set volume label */
36: #define FIOATTRIBSET    35  /* set file attribute */
37: #define FIOCONTIG       36  /* allocate contiguous space */
38: #define FIOREADDIR      37  /* read a directory entry (POSIX) */
39: #define FIOFSTATGET_OLD 38  /* get file status info - legacy */
40: #define FIOUNMOUNT      39  /* unmount disk volume */
41: #define FIOSCSICOMMAND  40  /* issue a SCSI command */
42: #define FIONCONTIG      41  /* get size of max contig area on dev */
43: #define FIOTRUNC        42  /* truncate file to specified length */
44: #define FIOGETFL        43  /* get file mode, like fcntl(F_GETFL) */
45: #define FIOTIMESET      44  /* change times on a file for utime() */
46: #define FIOINODETONAME  45  /* given inode number, return filename*/
47: #define FIOFSTATFSGET   46  /* get file system status info */
48: #define FIOMOVE         47  /* move file, ala mv, (mv not rename) */
49:
50: /* 64-bit ioctl codes, "long long *" expected as 3rd argument */
51: #define FIOCHKDSK       48
52: #define FIOCONTIG64     49
53: #define FIONCONTIG64    50
54: #define FIONFREE64      51
55: #define FIONREAD64      52
56: #define FIOSEEK64       53
57: #define FIOWHERE64      54
58: #define FIOTRUNC64      55
59:
60: #define FIOCOMMITFS     56
61: #define FIODATASYNC     57  /* sync to I/O data integrity */
62: #define FIOLINK         58  /* create a link */
63: #define FIOUNLINK       59  /* remove a link */
64: #define FIOACCESS       60  /* support POSIX access() */
65: #define FIOPATHCONF     61  /* support POSIX pathconf() */
66: #define FIOFCNTL        62  /* support POSIX fcntl() */
67: #define FIOCHMOD        63  /* support POSIX chmod() */
68: #define FIOFSTATGET     64  /* get stat info - POSIX  */
69: #define FIOUPDATE       65  /* update dosfs default create option */
70:
71: /* These are for HRFS but can be used for other future file systems */
72: #define FIOCOMMITPOLICYGETFS 66 /* get the file system commit policy */
73: #define FIOCOMMITPOLICYSETFS 67 /* set the file system commit policy */
74: #define FIOCOMMITPERIODGETFS 68 /* get the file system   periodic commit inte
75: #define FIOCOMMITPERIODSETFS 69 /* set the file system
76: #define FIOFSTATFSGET64      70 /* get file system status info */
```