NAME

mtio – general magnetic tape interface

SYNOPSIS

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

DESCRIPTION

1/2″, 1/4″, 4mm, and 8mm magnetic tape drives all share the same general character device interface.

There are two types of tape records: data records and end-of-file (EOF) records. EOF records are also known as tape marks and file marks. A record is separated by inter-record (or tape) gaps on a tape.

End-of-recorded-media (EOM) is indicated by two EOF marks on 1/2″ tape; by one on 1/4″, 4mm, and 8mm cartridge tapes.

1/2″ Reel Tape

Data bytes are recorded in parallel onto the 9-track tape. Since it is a variable-length tape device, the number of bytes in a physical record may vary.

The recording formats available (check specific tape drive) are 800 BPI, 1600 BPI, 6250 BPI, and data compression. Actual storage capacity is a function of the recording format and the length of the tape reel. For example, using a 2400 foot tape, 20 Mbyte can be stored using 800 BPI, 40 Mbyte using 1600 BPI, 140 Mbyte using 6250 BPI, or up to 700 Mbyte using data compression.

1/4″ Cartridge Tape

Data is recorded serially onto 1/4″ cartridge tape. The number of bytes per record is determined by the physical record size of the device. The I/O request size must be a multiple of the physical record size of the device. For QIC-11, QIC-24, and QIC-150 tape drives, the block size is 512 bytes.

The records are recorded on tracks in a serpentine motion. As one track is completed, the drive switches to the next and begins writing in the opposite direction, eliminating the wasted motion of rewinding. Each file, including the last, ends with one file mark.

Storage capacity is based on the number of tracks the drive is capable of recording. For example, 4-track drives can only record 20 Mbyte of data on a 450 foot tape; 9-track drives can record up to 45 Mbyte of data on a tape of the same length. QIC-11 is the only tape format available for 4-track tape drives. In contrast, 9-track tape drives can use either QIC-24 or QIC-11. Storage capacity is not appreciably affected by using either format. QIC-24 is preferable to QIC-11 because it records a reference signal to mark the position of the first track on the tape, and each block has a unique block number.

The QIC-150 tape drives require DC-6150 (or equivalent) tape cartridges for writing. However, they can read other tape cartridges in QIC-11, QIC-24, or QIC-120 tape formats.

8mm Cartridge Tape

Data is recorded serially onto 8mm helical scan cartridge tape. Since it is a variable-length tape device, the number of bytes in a physical record may vary. The recording formats available (check specific tape drive) are standard 2Gbyte, 5Gbyte, and compressed format.

4mm DAT Tape

Data is recorded either in Digital Data Storage (DDS) tape format or in Digital Data Storage, Data Compressed (DDS-DC) tape format. Since it is a variable-length tape device, the number of bytes in a physical record may vary. The recording formats available are standard 2Gbyte and compressed format.

Read Operation

read(2) reads the next record on the tape. The record size is passed back as the number of bytes read, provided it is no greater than the number requested. When a tape mark or end of data is read, a zero byte count is returned; another read will return an error. This is different from the older BSD behavior where another read will fetch the first record of the next tape file. If this behavior is required, device names containing the letter **b** (for BSD behavior) in the final component should be used.

Two successive reads returning zero byte counts indicate the EOM. No further reading should be performed past the EOM.

Fixed-length I/O tape devices require the number of bytes read to be a multiple of the physical record size. For example, 1/4″ cartridge tape devices only read multiples of 512 bytes. If the blocking factor is greater than 64512 bytes (minphys limit), fixed-length I/O tape devices read multiple records.

Most tape devices which support variable-length I/O operations may read a range of 1 to 65,535 bytes. If the record size exceeds 65,535 bytes, the driver reads multiple records to satisfy the request. These multiple records are limited to 65,534 bytes. Newer variable-length tape drivers may relax the above limitation and allow applications to read record sizes larger than 65,534. Refer to the specific tape driver man page for details.

Reading past logical EOT is transparent to the user. A read operation should never hit physical EOT.

Read requests that are lesser than a physical tape record are not allowed. Appropriate error is returned.

Write Operation

write(2) writes the next record on the tape. The record has the same length as the given buffer.

Writing is allowed on 1/4″ tape at either the beginning of tape or after the last written file on the tape.

Writing is not so restricted on 1/2″, 4mm, and 8mm cartridge tape. Care should be used when appending files onto 1/2″ reel tape devices, since an extra file mark is appended after the last file to mark the EOM. This extra file mark must be overwritten to prevent the creation of a null file. To facilitate write append operations, a space to the EOM ioctl is provided. Care should be taken when overwriting records; the erase head is just forward of the write head and any following records will also be erased.

Fixed-length I/O tape devices require the number of bytes written to be a multiple of the physical record size. For example, 1/4″ cartridge tape devices only write multiples of 512 bytes.

Fixed-length I/O tape devices write multiple records if the blocking factor is greater than 64,512 bytes (minphys limit). These multiple writes are limited to 64,512 bytes. For example, if a write request is issued for 65,536 bytes using a 1/4" cartridge tape, two writes are issued; the first for 64,512 bytes and the second for 1024 bytes.

Most tape devices which support variable-length I/O operations may write a range of 1 to 65,535 bytes. If the record size exceeds 65,535 bytes, the driver writes multiple records to satisfy the request. These multiple records are limited to 65534 bytes. As an example, if a write request for 65540 bytes is issued, two records are written; one for 65,534 bytes followed by another record for 6 byte. Newer variable-length tape drivers may relax the above limitation and allow applications to write record sizes larger than 65,534. Refer to the specific tape driver man page for details.

When logical EOT is encountered, a zero byte count is returned. The next write will complete successfully and the full byte count is returned. Another write will return a zero byte count. This allows the flushing of buffers. However, it is strongly recommended to terminate the writing and close the file as soon as possible.

Seeks are ignored in tape I/O.

Close Operation

Magnetic tapes are rewound when closed, except when the "no-rewind" devices have been specified. The names of no-rewind device files use the letter **n** as the end of the final component. The no-rewind version of **/dev/rmt/0l** is **/dev/0ln**. In case of error for a no-rewind device, the next open rewinds the device.

If the driver was opened for reading and a no-rewind device has been specified, the close advances the tape past the next filemark (unless the current file position is at EOM) leaving the tape correctly positioned to read the first record of the next file. However, if the tape is at the first record of a file it doesn't advance again to the first record of the next file. These semantics are different from the older BSD behavior. If BSD behavior is required where no implicit space operation is executed on close, the non-rewind device name containing the letter **b** (for BSD behavior) in the final component should be specified.

If data was written, a file mark is automatically written by the driver upon close. If the rewinding device was specified, the tape will be rewound after the file mark is written. If the user wrote a file mark prior to closing, then no file mark is written upon close. If a file positioning ioctl, like rewind, is issued after writing, a file mark is written before repositioning the tape.

All buffers are flushed on closing a tape device. Hence, one should check the value returned by the close operation.

Note that for 1/2" reel tape devices, two file marks are written to mark the EOM before rewinding or performing a file positioning ioctl. If the user wrote a file mark before closing a 1/2" reel tape device, the driver will always write a file mark before closing to insure that the end of recorded media is marked properly. If the non-rewinding device was specified, two file marks are written and the tape is left positioned between the two so that the second one is overwritten on a subsequent **open**(2) and **write**(2).

If no data was written and the driver was opened for WRITE-ONLY access, one or two file marks are written, thus creating a null file.

IOCTLS

Not all devices support all ioctls. The driver returns an **ENOTTY** error on unsupported ioctls.

The following structure definitions for magnetic tape ioctl commands are from **<sys/mtio.h>**:

The minor device byte structure looks as follows:

15 7	6	5	4	3	2	1 0
Unit # Bits 7-15	BSD behavior	Reserved	Density Select	Density Select	No rewind on Close	Unit # Bits 0-1

```
/*
 * Layout of minor device byte:
 */
#define MTUNIT(dev)          (((minor(dev) & 0xff80) >> 5) +(minor(dev) & 0x3))
#define MT_NOREWIND          (1 <<2)
#define MT_DENSITY_MASK      (3 <<3)
#define MT_DENSITY1          (0 <<3)          /* Lowest density/format */
#define MT_DENSITY2          (1 <<3)
#define MT_DENSITY3          (2 <<3)
#define MT_DENSITY4          (3 <<3)          /* Highest density/format */
#define MTMINOR(unit)        (((unit & 0x7fc) << 5) + (unit & 0x3))
#define MT_BSD               (1 <<6)          /* BSD behavior on close */

/* structure for MTIOCTOP – magnetic tape operation command */
struct   mtop {
         short               mt_op;           /* operation */
         daddr_t             mt_count;        /* number of operations */
};
```

The following operations of **MTIOCTOP** ioctl are supported:

MTWEOF write an end-of-file record

MTFSF forward space over file mark

MTBSF backward space over file mark (1/2", 8mm only)

MTFSR forward space to inter-record gap

MTBSR backward space to inter-record gap

MTREW rewind

MTOFFL rewind and take the drive off-line

MTNOP no operation, sets status only

MTRETEN	retension the tape (cartridge tape only)
MTERASE	erase the entire tape and rewind
MTEOM	position to EOM
MTNBSF	backward space file to beginning of file
MTSRSZ	set record size
MTGRSZ	get record size
MTLOAD	load the next tape cartridge into the tape drive.

```
/* structure for MTIOCGET – magnetic tape get status command */
struct   mtget {
         short          mt_type;        /* type of magtape device */
/* the following two registers are device dependent */
         short          mt_dsreg;       /* "drive status" register */
         short          mt_erreg;       /* "error" register */
/* optional error info. */
         daddr_t        mt_resid;       /* residual count */
         daddr_t        mt_fileno;      /* file number of current position */
         daddr_t        mt_blkno;       /* block number of current position */
         u_short        mt_flags;
         short          mt_bf;          /* optimum blocking factor */
};
```

When spacing forward over a record (either data or EOF), the tape head is positioned in the tape gap between the record just skipped and the next record. When spacing forward over file marks (EOF records), the tape head is positioned in the tape gap between the next EOF record and the record that follows it.

When spacing backward over a record (either data or EOF), the tape head is positioned in the tape gap immediately preceding the tape record where the tape head is currently positioned. When spacing backward over file marks (EOF records), the tape head is positioned in the tape gap preceding the EOF. Thus the next read would fetch the EOF.

Record skipping does not go past a file mark; file skipping does not go past the EOM. After an **MTFSR** <huge number> command the driver leaves the tape logically positioned *before* the EOF. A related feature is that EOFs remain pending until the tape is closed. For example, a program which first reads all the records of a file up to and including the EOF and then performs an MTFSF command will leave the tape positioned just after that same EOF, rather than skipping the next file.

The **MTNBSF** and **MTFSF** operations are inverses. Thus, an “**MTFSF** −1” is equivalent to an “**MTNBSF** 1”. An “**MTNBSF** 0” is the same as “**MTFSF** 0”; both position the tape device at the beginning of the current file.

MTBSF moves the tape backwards by file marks. The tape position will end on the beginning of tape side of the desired file mark.

MTBSR and **MTFSR** operations perform much like space file operations, except that they move by records instead of files. Variable-length I/O devices (1/2″ reel, for example) space actual records; fixed-length I/O devices space physical records (blocks). 1/4″ cartridge tape, for example, spaces 512 byte physical records. The status ioctl residual count contains the number of files or records not skipped.

MTOFFL rewinds and, if appropriate, takes the device off-line by unloading the tape. It is recommended that the device be closed after offlining and then re-opened after a tape has been inserted to facilitate portability to other platforms and other operating systems. Attempting to re-open the device with no tape will result in an error unless the **O_DELAY** flag is used. (See **open**(2).)

The **MTRETEN** retension ioctl applies only to 1/4″ cartridge tape devices. It is used to restore tape tension, improving the tape's soft error rate after extensive start-stop operations or long-term storage.

MTERASE rewinds the tape, erases it completely, and returns to the beginning of tape.

MTEOM positions the tape at a location just after the last file written on the tape. For 1/4″ cartridge and 8mm tape, this is after the last file mark on the tape. For 1/2″ reel tape, this is just after the first file mark but before the second (and last) file mark on the tape. Additional files can then be appended onto the tape from that point.

Note the difference between **MTBSF** (backspace over file mark) and **MTNBSF** (backspace file to beginning of file). The former moves the tape backward until it crosses an EOF mark, leaving the tape positioned *before* the file mark. The latter leaves the tape positioned *after* the file mark. Hence, "**MTNBSF** n" is equivalent to "**MTBSF** (n+1)" followed by "**MTFSF** 1". The 1/4 ″ cartridge tape devices do not support **MTBSF.**

MTSRSZ and **MTGRSZ** are used to set and get fixed record lengths. The **MTSRSZ** ioctl allows variable length and fixed length tape drives that support multiple record sizes to set the record length. The **mt_count** field of the **mtop** struct is used to pass the record size to/from the **st** driver. A value of **0** indicates variable record size. The **MTSRSZ** ioctl makes a variable-length tape device behave like a fixed-length tape device. Refer to the specific tape driver man page for details.

The **MTIOCGET** get status ioctl call returns the drive ID (*mt_type*), sense key error (*mt_erreg*), file number (*mt_fileno*), optimum blocking factor (*mt_bf*) and record number (*mt_blkno*) of the last error. The residual count (*mt_resid*) is set to the number of bytes not transferred or files/records not spaced. The flags word (*mt_flags*) contains information such as whether the device is SCSI, whether it is a reel device and whether the device supports absolute file positioning.

MTLOAD loads the next tape cartridge into the tape drive. This is generally only used with stacker and tower type tape drives which handle multiple tapes per tape drive. A tape device without a tape inserted can be opened with the **O_DELAY** flag, in order to execute this operation.

EXAMPLES

Suppose you have written three files to the non-rewinding 1/2″ tape device, **/dev/rmt/0ln,** and that you want to go back and **dd**(1M) the second file off the tape. The commands to do this are:

```
mt –f /dev/rmt/0ln bsf 3
mt –f /dev/rmt/0ln fsf 1
dd if=/dev/rmt/0ln
```

To accomplish the same tape positioning in a C program, followed by a get status ioctl:

```
struct mtop mt_command;
struct mtget mt_status;

mt_command.mt_op = MTBSF;
mt_command.mt_count = 3;
ioctl(fd, MTIOCTOP, &mt_command);
mt_command.mt_op = MTFSF;
mt_command.mt_count = 1;
ioctl(fd, MTIOCTOP, &mt_command);
ioctl(fd, MTIOCGET, (char *)&mt_status);
```

or

```
mt_command.mt_op = MTNBSF;
mt_command.mt_count = 2;
ioctl(fd, MTIOCTOP, &mt_command);
ioctl(fd, MTIOCGET, (char *)&mt_status);
```

FILES

/dev/rmt/<*unit number*><*density*>[<*BSD behavior*>][<*no rewind*>]

density	**l, m, h, u/c** (low, medium, high, ultra/compressed, respectively)
BSD behavior (optional)	**b**
no rewind (optional)	**n**

For example, **/dev/rmt/0hbn** specifies unit 0, high density, BSD behavior and no rewind.

SEE ALSO

mt(1), **tar**(1), **dd**(1M), **read**(2), **write**(2), **ar**(4), **st**(7D)

1/4 Inch Tape Drive Tutorial