NAME

fdio – floppy disk control operations

SYNOPSIS

#include <sys/fdio.h>

DESCRIPTION

The Solaris floppy driver supports a set of **ioctl**(2) requests for getting and setting the floppy drive characteristics. Basic to these **ioctl( )** requests are the definitions in **<sys/fdio.h>**.

IOCTLS

The following **ioctl( )** requests are available only on the Solaris floppy driver.

FDDEFGEOCHAR
x86: This **ioctl( )** forces the floppy driver to restore the diskette and drive characteristics and geometry, and partition information to default values based on the device configuration.

FDGETCHANGE
The argument is a pointer to an **int.** This **ioctl( )** returns the status of the diskette-changed signal from the floppy interface. The following defines are provided for cohesion.

Note that for x86 systems, **FDGC_DETECTED** (which is available only on x86) should be used instead of **FDGC_HISTORY**.

```
/*
 * Used by FDGETCHANGE, returned state of the sense disk change bit.
 */
#define FDGC_HISTORY      0x01   /* disk has changed since last call */
#define FDGC_CURRENT      0x02   /* current state of disk change */
#define FDGC_CURWPROT     0x10   /* current state of write protect */
#define FDGC_DETECTED     0x20   /* previous state of DISK CHANGE */
```

FDIOGCHAR
The argument is a pointer to an **fd_char** structure (described below). This **ioctl( )** gets the characteristics of the floppy diskette from the floppy controller.

FDIOSCHAR
The argument is a pointer to an **fd_char** structure (described below). This **ioctl( )** sets the characteristics of the floppy diskette for the floppy controller. Typical values in the fd_char structure for a high density diskette:

field	value	
fdc_medium	0	
fdc_transfer_rate	500	
fdc_ncyl	80	
fdc_nhead	2	
fdc_sec_size	512	
fdc_secptrack	18	
fdc_steps	-1	{ This field doesn’t apply. }

```
/*
 * Floppy characteristics
 */
struct fd_char {
        u_char   fdc_medium;         /* equals 1 if medium type */
        int      fdc_transfer_rate;  /* transfer rate */
        int      fdc_ncyl;           /* number of cylinders */
        int      fdc_nhead;          /* number of heads */
        int      fdc_sec_size;       /* sector size */
        int      fdc_secptrack;      /* sectors per track */
        int      fdc_steps;          /* no. of steps per data track */
};
```

FDGETDRIVECHAR
The argument to this **ioctl( )** is a pointer to an **fd_drive** structure (described below). This **ioctl( )** gets the characteristics of the floppy drive from the floppy controller.

FDSETDRIVECHAR
x86: The argument to this **ioctl( )** is a pointer to an **fd_drive** structure (described below). This **ioctl( )** sets the characteristics of the floppy drive for the floppy controller. Only **fdd_steprate**, **fdd_headsettle**, **fdd_motoron**, and **fdd_motoroff** are actually used by the floppy disk driver.

```
/*
 * Floppy Drive characteristics
 */
struct fd_drive {
        int   fdd_ejectable;      /* does the drive support eject? */
        int   fdd_maxsearch;      /* size of per-unit search table */
        int   fdd_writeprecomp;   /* cyl to start write prcompensation */
        int   fdd_writereduce;    /* cyl to start recucing write current */
        int   fdd_stepwidth;      /* width of step pulse in 1 us units */
        int   fdd_steprate;       /* step rate in 100 us units */
        int   fdd_headsettle;     /* delay, in 100 us units */
        int   fdd_headload;       /* delay, in 100 us units */
        int   fdd_headunload;     /* delay, in 100 us units */
        int   fdd_motoron;        /* delay, in 100 ms units */
        int   fdd_motoroff;       /* delay, in 100 ms units */
        int   fdd_precomplevel;   /* bit shift, in nano-secs */
        int   fdd_pins;           /* defines meaning of pin 1, 2, 4 and 34 */
        int   fdd_flags;          /* TRUE READY, Starting Sector #, & Motor On */
};
```

FDGETSEARCH Not available.

FDSETSEARCH Not available.

FDEJECT SPARC: This **ioctl()** requests the floppy drive to eject the diskette.

FDIOCMD The argument is a pointer to an **fd_cmd** structure (described below). This **ioctl( )** allows access to the floppy diskette using the floppy device driver. Only the **FDCMD_WRITE**, **FDCMD_READ**, and **FDCMD_FORMAT_TR** commands are currently available.

```
struct fd_cmd {
        u_short   fdc_cmd;         /* command to be executed */
        int       fdc_flags;       /* execution flags (x86 only) */
        daddr_t   fdc_blkno;       /* disk address for command */
        int       fdc_secnt;       /* sector count for command */
        caddr_t   fdc_bufaddr;     /* user's buffer address */
        u_int     fdc_buflen;      /* size of user's buffer */
};

/*
 * Floppy commands
 */
#define    FDCMD_WRITE            1
#define    FDCMD_READ             2
#define    FDCMD_SEEK             3
#define    FDCMD_REZERO           4
#define    FDCMD_FORMAT_UNIT      5
#define    FDCMD_FORMAT_TRACK     6
```

FDRAW The argument is a pointer to an **fd_raw** structure (described below). This **ioctl( )** allows direct control of the floppy drive using the floppy controller. Refer to the appropriate floppy-controller data sheet for full details on required command bytes and returned result bytes. The following commands are supported.

```
/*
* Floppy raw commands
*/
#define FDRAW_SPECIFY      0x03
#define FDRAW_READID       0x0a      (x86 only)
#define FDRAW_SENSE_DRV    0x04
#define FDRAW_REZERO       0x07
#define FDRAW_SEEK         0x0f
#define FDRAW_SENSE_INT    0x08      (x86 only)
#define FDRAW_FORMAT       0x0d
#define FDRAW_READTRACK    0x02
#define FDRAW_WRCMD        0x05
#define FDRAW_RDCMD        0x06
#define FDRAW_WRITEDEL     0x09
#define FDRAW_READDEL      0x0c
```

```
/*
 * Used by FDRAW
 */
struct fd_raw {
        char    fdr_cmd[10];     /* user-supplied command bytes */
        short   fdr_cnum;        /* number of command bytes */
        char    fdr_result[10];  /* controller-supplied result bytes */
        short   fdr_nbytes;      /* number to transfer if read/write command */
        char    *fdr_addr;       /* where to transfer if read/write command */
};
```

SEE ALSO

ioctl(2), **dkio**(7I), **fd**(7D), **hdio**(7I)