NAME

fsdb_ufs – ufs file system debugger

SYNOPSIS

fsdb –F ufs [*generic_options*] [*specific_options*] *special*

AVAILABILITY

SUNWxxx

DESCRIPTION

The **fsdb_ufs** command is an interactive tool that can be used to patch up a damaged UFS file system. It has conversions to translate block and i-numbers into their corresponding disk addresses. Also included are mnemonic offsets to access different parts of an inode. These greatly simplify the process of correcting control block entries or descending the file system tree.

fsdb contains several error-checking routines to verify inode and block addresses. These can be disabled if necessary by invoking **fsdb** with the –**o** option or by the use of the **o** command.

fsdb reads a block at a time and will therefore work with raw as well as block I/O devices. A buffer management routine is used to retain commonly used blocks of data in order to reduce the number of read system calls. All assignment operations result in an immediate write-through of the corresponding block. Note that in order to modify any portion of the disk, **fsdb** must be invoked with the **w** option.

Wherever possible, **adb-**like syntax was adopted to promote the use of **fsdb** through familiarity.

OPTIONS

The following option is supported:

–**o** Specify UFS file system specific options. These options can be any combination of the following separated by commas (with no intervening spaces). The options available are:

?	Display usage
o	Override some error conditions
p='string'	set prompt to string
w	open for write

USAGE

Numbers are considered hexadecimal by default. However, the user has control over how data is to be displayed or accepted. The **base** command will display or set the input/output base. Once set, all input will default to this base and all output will be shown in this base. The base can be overriden temporarily for input by preceding hexadecimal numbers with '**0x**', preceding decimal numbers with '**0t**', or octal numbers with '**0**'. Hexadecimal numbers beginning with **a-f** or **A-F** must be preceded with '**0x**' to distinguish them from commands.

Disk addressing by **fsdb** is at the byte level. However, **fsdb** offers many commands to convert a desired inode, directory entry, block, superblock etc. to a byte address. Once the address has been calculated, **fsdb** will record the result in **dot** (see next paragraph).

Several global values are maintained by **fsdb**:

- the current base (referred to as **base**),
- the current address (referred to as **dot**),
- the current inode (referred to as **inode**),
- the current count (referred to as **count**),
- and the current type (referred to as **type**).

Most commands use the preset value of **dot** in their execution. For example,

> 2:inode

will first set the value of **dot** to 2, ':', will alert the start of a command, and the **inode** command will set **inode** to 2. A count is specified after a ','. Once set, **count** will remain at this value until a new command is encountered which will then reset the value back to 1 (the default). So, if

> 2000,400/X

is typed, 400 hex longs are listed from 2000, and when completed, the value of **dot** will be **2000 + 400 * sizeof (long)**. If a carriage-return is then typed, the output routine will use the current values of **dot**, **count**, and **type** and display 400 more hex longs. A '*' will cause the entire block to be displayed.

End of fragment, block and file are maintained by **fsdb**. When displaying data as fragments or blocks, an error message will be displayed when the end of fragment or block is reached. When displaying data using the **db**, **ib**, **directory**, or **file** commands an error message is displayed if the end of file is reached. This is mainly needed to avoid passing the end of a directory or file and getting unknown and unwanted results.

An example showing several commands and the use of carriage-return would be:

> 2:ino; 0:dir?d
or
> 2:ino; 0:db:block?d

The two examples are synonymous for getting to the first directory entry of the root of the file system. Once there, subsequent carriage-returns (or +, -) will advance to subsequent entries. Note that

> 2:inode; :ls
or
> :ls /

is again synonymous.

Expressions

The symbols recognized by **fsdb** are:

CARRIAGE-RETURN
: update the value of **dot** by the current value of **type** and display using the current value of **count**.

#
: numeric expressions may be composed of +, -, *, and % operators (evaluated left to right) and may use parentheses. Once evaluated, the value of **dot** is updated.

, *count* count indicator. The global value of **count** will be updated to **count**. The value of **count** will remain until a new command is run. A count specifier of '*' will attempt to show a *blocks's* worth of information. The default for **count** is 1.

? *f* display in structured style with format specifier *f* (see **Formatted Output** section).

/ *f* display in unstructured style with format specifier *f* (see **Formatted Output** section).

. the value of **dot**.

+*e* increment the value of **dot** by the expression *e.* The amount actually incremented is dependent on the size of **type**:

dot = dot + e * sizeof (type)

The default for *e* is **1**.

-*e* decrement the value of **dot** by the expression *e* (see +).

*****e* multiply the value of **dot** by the expression *e.* Multiplication and division don't use **type**. In the above calculation of **dot**, consider the **sizeof(type)** to be **1**.

%*e* divide the value of **dot** by the expression *e* (see *).

< *name* restore an address saved in register *name*. *name* must be a single letter or digit.

> *name* save an address in register *name*. *name* must be a single letter or digit.

= *f* display indicator. If *f* is a legitimate format specifier (see **Formatted Output** section), then the value of **dot** is displayed using format specifier *f*. Otherwise, assignment is assumed (see next item).

= [*s*] [*e*] assignment indicator. The address pointed to by **dot** has its contents changed to the value of the expression *e* or to the ASCII representation of the quoted (") string *s*. This may be useful for changing directory names or ASCII file information.

=+ *e* incremental assignment. The address pointed to by **dot** has its contents incremented by expression *e*.

=- *e* decremental assignment. The address pointed to by **dot** has its contents decremented by expression *e*.

Commands

A command must be prefixed by a ':' character. Only enough letters of the command to uniquely distinguish it are needed. Multiple commands may be entered on one line by separating them by a space, tab or ';'.

In order to view a potentially unmounted disk in a reasonable manner, **fsdb** offers the *cd*, *pwd*, *ls* and *find* commands. The functionality of these commands substantially matches those of its UNIX counterparts (see individual command for details). The '*', '?', and '[-]' wild card characters are available.

base=b display or set base. As stated above, all input and output is governed by the current **base**. If the '=b' is left off, the current **base** is displayed. Otherwise, the current **base** is set to *b*. Note that this is interpreted using the old value of **base**, so to ensure correctness use the '0', '0t', or '0x' prefix when changing the **base**. The default for **base** is hexadecimal.

block convert the value of **dot** to a block address.

cd *dir* change the current directory to directory *dir*. The current values of **inode** and **dot** are also updated. If no *dir* is specified, then change directories to inode **2** ("/").

cg convert the value of **dot** to a cylinder group.

directory If the current **inode** is a directory, then the value of **dot** is converted to a directory slot offset in that directory and **dot** now points to this entry.

file the value of **dot** is taken as a relative block count from the beginning of the file. The value of **dot** is updated to the first byte of this block.

find *dir* [–**name** *n*] [–**inum** *i*]
find files by name or i-number. **find** recursively searches directory **dir** and below for filenames whose i-number matches *i* or whose name matches pattern *n*. Note that only one of the two options (-name or -inum) may be used at one time. Also, the -print is not needed or accepted.

fill=*p* fill an area of disk with pattern *p*. The area of disk is delimited by **dot** and **count**.

fragment convert the value of *dot* to a fragment address. The only difference between the **fragment** command and the **block** command is the amount that is able to be displayed.

inode convert the value of *dot* to an inode address. If successful, the current value of **inode** will be updated as well as the value of *dot*. As a convenient shorthand, if ':inode' appears at the beginning of the line, the value of *dot* is set to the current **inode** and that inode is displayed in inode format.

ls [–**R**] [–**l**] *pat1 pat2* . . .
list directories or files. If no file is specified, the current directory is assumed. Either or both of the options may be used (but, if used, *must* be specified before the filename specifiers). Also, as stated above, wild card characters are available and multiple arguments may be given. The long listing shows only the i-number and the name; use the **inode** command with '?i' to get more information.

override toggle the value of override. Some error conditions may be overriden if override is toggled on.

prompt *p* change the **fsdb** prompt to *p*. *p* must be surrounded by (")s.

pwd display the current working directory.

quit quit **fsdb**.

sb the value of *dot* is taken as a cylinder group number and then converted to the address of the superblock in that cylinder group. As a shorthand, ':sb' at the beginning of a line will set the value of *dot* to *the* superblock and display it in superblock format.

! escape to shell

Inode Commands

In addition to the above commands, there are several commands that deal with inode fields and operate directly on the current **inode** (they still require the ':'). They may be used to more easily display or change the particular fields. The value of *dot* is only used by the '**:db**' and '**:ib**' commands. Upon completion of the command, the value of *dot* is changed to point to that particular field. For example,

> :ln=+1

would increment the link count of the current **inode** and set the value of *dot* to the address of the link count field.

at access time.

bs block size.

ct creation time.

db use the current value of *dot* as a direct block index, where direct blocks number from 0 - 11. In order to display the block itself, you need to 'pipe' this result into the **block** or **fragment** command. For example,
> 1:db:block,20/X
would get the contents of data block field 1 from the inode and convert it to a block address. 20 longs are then displayed in hexadecimal (see **Formatted Output** sub-section).

gid group id.

ib use the current value of *dot* as an indirect block index where indirect blocks number from 0 - 2. This will only get the indirect block itself (the block containing the pointers to the actual blocks). Use the **file** command and start at block 12 to get to the actual blocks.

ln link count.

mt modification time.

md mode.

maj major device number.

min minor device number.

nm although listed here, this command actually operates on the directory name field. Once poised at the desired directory entry (using the *directory* command), this command will allow you to change or display the directory name. For example,

> 7:dir:nm="foo"

will get the **7**th directory entry of the current **inode** and change its name to foo. Note that names cannot be made larger than the field is set up for. If an attempt is made, the string is truncated to fit and a warning message to this effect is displayed.

sz file size.

uid user id.

Formatted Output

There are two styles and many format types. The two styles are structured and unstructured. Structured output is used to display inodes, directories, superblocks and the like. Unstructured just displays raw data. The following table shows the different ways of displaying:

?

c	display as cylinder groups
i	display as inodes
d	display as directories
s	display as superblocks

/

b	display as bytes
c	display as characters
o O	display as octal shorts or longs
d D	display as decimal shorts or longs
x X	display as hexadecimal shorts or longs

The format specifier immediately follows the ′/′ or ′?′ character. The values displayed by ′/b′ and all ′?′ formats are displayed in the current **base**. Also, **type** is appropriately updated upon completion.

EXAMPLES

> 2000+400%(20+20)=D
will display **2010** in decimal (use of **fsdb** as a calculator for complex arithmetic).

> 386:ino?i display i-number **386** in an inode format. This now becomes the current **inode**.

> :ln=4 changes the link count for the current **inode** to **4**.

> :ln=+1 increments the link count by **1**.

> :ct=X display the creation time as a hexadecimal long.

> :mt=t display the modification time in time format.

> 0:file/c displays, in ASCII, block zero of the file associated with the current **inode**.

> 2:ino,∗?d displays the first blocks worth of directory entries for the root inode of this file system. It will stop prematurely if the EOF is reached.

> 5:dir:inode; 0:file,∗/c
changes the current inode to that associated with the **5**th directory entry (numbered from zero) of the current **inode.** The first logical block of the file is then displayed in ASCII.

> :sb displays the superblock of this file system.

> 1:cg?c displays cylinder group information and summary for cylinder group **1**.

> 2:inode; 7:dir=3 changes the i-number for the seventh directory slot in the root directory to **3**.

> 7:dir:nm="name" changes the name field in the directory slot to *name*.

> 2:db:block,∗?d displays the third block of the current **inode** as directory entries.

> 3c3:fragment,20:fill=0x20
get fragment **3c3** and fill **20 type** elements with **0x20**.

> 2050=0xffff set the contents of address **2050** to **0xffffffff**. **0xffffffff** may be truncated depending on the current **type**.

> 1c92434="this is some text"
will place the ASCII for the string at **1c92434**.

SEE ALSO

clri(1M), **fsck_ufs**(1M), **dir_ufs**(4), **fs_ufs**(4)

WARNINGS

Since **fsdb** reads the disk raw, extreme caution is advised in determining its availability of **fsdb** on the system. Suggested permissions are 600 and owned by bin.

NOTES

The old command line syntax for clearing i-nodes using the ufs-specific **'-z i-number'** option is still supported by the new debugger, though it is obsolete and will be removed in a future release. Use of this flag will result in correct operation, but an error message will be printed warning of the impending obsolesence of this option to the command. The equivalent functionality is available using the more flexible **clri**(1M) command.