NAME

cpio – copy file archives in and out

SYNOPSIS

cpio –i [**bBcdfkmrsStuvV6**] [**–C** *bufsize*] [**–E** *file*] [**–H** *header*]
[**–I** *file* [**–M** *message*]] [**–R** *id*] [*pattern* . . .]
cpio –o [**aABcLvV**] [**–C** *bufsize*] [**–H** *header*] [**–O** *file* [**–M** *message*]]
cpio –p [**adlLmuvV**] [**–R** *id*] *directory*

AVAILABILITY

SUNWcsu

DESCRIPTION

The **cpio** command copies files in to and out from a **cpio** archive. The **cpio** achive may span multiple volumes. The **–i**, **–o**, and **–p** options select the action to be performed. The following list describes each of the actions (which are mutually exclusive).

Copy In Mode

cpio –i (copy in) extracts files from the standard input, which is assumed to be the product of a previous **cpio –o**. Only files with names that match *patterns* are selected. See **sh**(1) and **OPERANDS** for more information about *pattern*. Extracted files are conditionally created and copied into the current directory tree based on the options described below. The permissions of the files will be those of the previous **cpio –o**. Owner and group will be the same as the current user unless the current user is super-user. If this is true, owner and group will be the same as those resulting from the previous **cpio –o**. Note that if **cpio –i** tries to create a file that already exists and the existing file is the same age or younger (**newer**), **cpio** will output a warning message and not replace the file. (The **–u** option can be used to overwrite, unconditionally, the existing file.)

Copy Out Mode

cpio –o (copy out) reads the standard input to obtain a list of path names and copies those files onto the standard output together with path name and status information. Output is padded to a 512-byte boundary by default or to the user specified block size (with the **–B** or **–C** options) or to some device-dependent block size where necessary (as with the CTC tape).

Pass Mode

cpio –p (pass) reads the standard input to obtain a list of path names of files that are conditionally created and copied into the destination *directory* tree based on the options described below.

Note: **cpio** assumes four-byte words.

If, when writing to a character device (**–o**) or reading from a character device (**–i**), **cpio** reaches the end of a medium (such as the end of a diskette), and the **–O** and **–I** options are not used, **cpio** prints the following message:

To continue, type device/file name when ready.

To continue, you must replace the medium and type the character special device name (**/dev/rdiskette** for example) and press RETURN. You may want to continue by directing **cpio** to use a different device. For example, if you have two floppy drives you may want to switch between them so **cpio** can proceed while you are changing the floppies. (Simply pressing RETURN causes the **cpio** process to exit.)

OPTIONS

The following options are supported:

–i (copy in) **cpio –i** extracts files from the standard input.

–o (copy out) **cpio –o** reads the standard input to obtain a list of path names and copies those files onto the standard output.

–p (pass) **cpio –p** reads the standard input to obtain a list of path names of files.

The following options can be appended in any sequence to the **–o**, **–i**, or **–p** options:

–a Reset access times of input files after they have been copied. Access times are not reset for linked files when **cpio –pla** is specified (mutually exclusive with **–m**).

–A Append files to an archive. The **–A** option requires the **–O** option. Valid only with archives that are files, or that are on floppy diskettes or hard disk partitions.

–b Reverse the order of the bytes within each word. (Use only with the **–i** option.)

–B Block input/output 5120 bytes to the record. The default buffer size is 512 bytes when this and the **–C** options are not used. **–B** does not apply to the *pass* option; **–B** is meaningful only with data directed to or from a character special device, for example, **/dev/rmt/0m**.

–c Read or write header information in ASCII character form for portability. Use this option between SVR4-based machines, or the **–H odc** option between unknown machines. The **–c** option implies the use of expanded device numbers, which are only supported on SVR4-based systems. When transferring files between Solaris 1.x or Interactive UNIX and Solaris 2.x use **–H odc**.

–C *bufsize* Block input/output *bufsize* bytes to the record, where *bufsize* is replaced by a positive integer. The default buffer size is 512 bytes when this and **–B** options are not used. (**–C** does not apply to the *pass* option; **–C** is meaningful only with data directed to or from a character special device, for example, **/dev/rmt/0m**.)

–d Create directories as needed.

–E *file* Specify an input file (*file*) that contains a list of filenames to be extracted from the archive (one filename per line).

–f Copy in all files except those in *patterns*. (See **OPERANDS** for a description of *patterns*.)

–H *header* Read or write header information in *header* format. Always use this option or the **–c** option when the origin and the destination machines are different types (mutually exclusive with **–c** and **–6**).

Valid values for *header* are:

bar	**bar** head and format. Used only with the –**i** option (read only)
crc | **CRC**	ASCII header with expanded device numbers and an additional per-file checksum
odc	ASCII header with small device numbers. This is the IEEE/P1003 Data Interchange Standard cpio header and format. It has the widest range of portability of any of the archive formats. It is the official format for transferring files between POSIX-conforming systems. Use this format to communicate with Solaris 1.x and Interactive UNIX.
tar | **TAR**	**tar** header and format
ustar | **USTAR**	IEEE/P1003 Data Interchange Standard tar header and format

–**I** *file* Read the contents of *file* as an input archive. If *file* is a character special device, and the current medium has been completely read, replace the medium and press RETURN to continue to the next medium. This option is used only with the –**i** option.

–**k** Attempt to skip corrupted file headers and I/O errors that may be encountered. If you want to copy files from a medium that is corrupted or out of sequence, this option lets you read only those files with good headers. (For **cpio** archives that contain other **cpio** archives, if an error is encountered **cpio** may terminate prematurely. **cpio** will find the next good header, which may be one for a smaller archive, and terminate when the smaller archive's trailer is encountered.) Used only with the –**i** option.

–**l** Whenever possible, link files rather than copying them. (Usable only with the –**p** option.)

–**L** Follow symbolic links. The default is not to follow symbolic links.

–**m** Retain previous file modification time. This option is ineffective on directories that are being copied (mutually exclusive with –**a**).

–**M** *message* Define a *message* to use when switching media. When you use the –**O** or –**I** options and specify a character special device, you can use this option to define the message that is printed when you reach the end of the medium. One **%d** can be placed in *message* to print the sequence number of the next medium needed to continue.

–**O** *file* Direct the output of **cpio** to *file.* If *file* is a character special device and the current medium is full, replace the medium and type a carriage return to continue to the next medium. Use only with the –**o** option.

–P Preserve ACLs. If the option is used for output, ACLs if existed are written along with other attributes to the standard output. ACLs are created as special files with a special file type. If the option is used for input, ACLs if existed are extracted along with other attributes from standard input. The option recognizes the special file type. Note that errors will occur if a cpio archive with ACLs is extracted by previous versions of **cpio**.

–r Interactively rename files. If the user types a carriage return alone, the file is skipped. If the user types a ''.'' the original pathname will be retained. (Not available with **cpio** –**p**.)

–R *id* Reassign ownership and group information for each file to *user ID* (*ID* must be a valid login ID from **/etc/passwd**). This option is valid only for the super-user.

–s Swap bytes within each half word.

–S Swap halfwords within each word.

–t Print a table of contents of the input. No files are created (mutually exclusive with –**V**).

–u Copy unconditionally (normally, an older file will not replace a newer file with the same name).

–v Verbose. Print a list of file names. When used with the –**t** option, the table of contents looks like the output of an **ls** –**l** command (see **ls**(1)).

–V Special verbose. Print a dot for each file read or written. Useful to assure the user that **cpio** is working without printing out all file names.

–6 Process a UNIX System Sixth Edition archive format file. Use only with the –**i** option (mutually exclusive with –**c** and –**H**)).

OPERANDS

The following operands are supported:

directory A path name of an existing directory to be used as the target of **cpio** –**p**.

pattern Expressions making use of a pattern-matching notation similar to that used by the shell (see **sh**(1)) for filename pattern matching, and similar to regular expressions. The following metacharacters are defined:

- * Matches any string, including the empty string.
- **?** Matches any single character.
- **[. . .]** Matches any one of the enclosed characters. A pair of characters separated by '–' matches any symbol between the pair (inclusive), as defined by the system default collating sequence. If the first character following the opening **'['** is a **'!'**, the results are unspecified.
- **!** means *not*. (For example, the **!abc*** pattern would exclude all files that begin with **abc**.)

In *patterns*, metacharacters **?**, *, and **[. . .]** match the slash (/) character, and backslash (\) is an escape character. Multiple cases of *pattern* can be specified and if no *pattern* is specified, the default for *pattern* is * (that is, select all files).

Each pattern must be enclosed in double quotes; otherwise, the name of a file in the current directory might be used.

EXAMPLES

The following examples show three uses of **cpio**.

When standard input is directed through a pipe to **cpio** −**o**, it groups the files so they can be directed (**>**) to a single file (**../newfile**). The −**c** option insures that the file will be portable to other machines (as would the −**H** option). Instead of **ls**(1), you could use **find**(1), **echo**(1), **cat**(1), and so on, to pipe a list of names to **cpio**. You could direct the output to a device instead of a file.

example% ls | cpio −oc > ../newfile

cpio −**i** uses the output file of **cpio** −**o** (directed through a pipe with **cat** in the example below), extracts those files that match the patterns (**memo/a1**, **memo/b***), creates directories below the current directory as needed (−**d** option), and places the files in the appropriate directories. The −**c** option is used if the input file was created with a portable header. If no patterns were given, all files from **newfile** would be placed in the directory.

example% cat newfile | cpio −icd "memo/a1" "memo/b*"

cpio −**p** takes the file names piped to it and copies or links (−**l** option) those files to another directory (**newdir** in the example below). The −**d** option says to create directories as needed. The −**m** option says retain the modification time. (It is important to use the −**depth** option of **find**(1) to generate path names for **cpio**. This eliminates problems **cpio** could have trying to create files under read-only directories.) The destination directory, **newdir**, must exist.

example% find . −depth −print | cpio −pdlmv newdir

Note that when you use **cpio** in conjunction with **find**, if you use the −**L** option with **cpio** then you must use the −**follow** option with **find** and vice versa. Otherwise there will be undesirable results.

Note that for multi-reel archives, dismount the old volume, mount the new one, and continue to the next tape by typing the name of the next device (probably the same as the first reel). To stop, type a RETURN and **cpio** will end.

ENVIRONMENT

See **environ**(5) for descriptions of the following environment variables that affect the execution of **cpio**: **LC_COLLATE**, **LC_CTYPE**, **LC_MESSAGES**, **LC_TIME**, **TZ**, and **NLSPATH**.

EXIT STATUS

The following exit values are returned:

0 Successful completion.
>0 An error occurred.

SEE ALSO

ar(1), **cat**(1), **echo**(1), **find**(1), **ls**(1), **setfacl**(1), **sh**(1), **tar**(1), **archives**(4), **environ**(5)

NOTES

Path names are restricted to 256 characters for the binary (the default) and –**H odc** header formats. Otherwise, path names are restricted to 1024 characters.

Only the super-user can copy special files.

Blocks are reported in 512-byte quantities.

If a file has **000** permissions, contains more than 0 characters of data, and the user is not root, the file will not be saved or restored.

The inode number stored in the header, (**/usr/include/archives.h**) is an unsigned short which is 2 bytes. This limits the range of inode numbers from **0** to **65535**. Files which are hard linked must fall in this inode range. This could be a problem when moving **cpio** archives between different vendors' machines.

When the Volume Management daemon is running, accesses to floppy devices through the conventional device names (eg, /dev/rdiskette) may not succeed. See **vold(1m)** for further details.