

NAME

zp – ZPAQ open standard maximum compressor with prebuilt compression levels

SYNOPSIS

```
create : zp cN archive.zpaq file [file ...]
append : zp aN archive.zpaq file [file ...]
list   : zp l archive.zpaq
extract: zp [ex][N] archive.zpaq
```

DESCRIPTION**General**

PAQ family was a series of open source data compression archivers that have evolved through collaborative development to top rankings on several benchmarks measuring compression ratio although at the expense of speed and memory usage.

ZPAQ ia a proposed standard format for highly compressed data that allows new compression algorithms to be developed without breaking compatibility with older programs. Zp is based on PAQ-like context mixing algorithms which are top ranked on many benchmarks. The format supports archivers, single file compressors, and memory to memory compression.

Zp’s goal is a high compression ratio in an open format without loss of compatibility between versions as advanced compression techniques are discovered.

If you compress in Windows and extract in Linux, then the program will change “\” to “/” during extraction and vice versa. Slashes can be stored with either convention. (The program guesses the operating system by counting “/” and “\” in the *PATH* environment variable. If this heuristic fails (*PATH* not defined) then no slash translation is done).

Paths must be relative to the current directory. The program will warn if you store an absolute path. You can only extract such files with command **e** or by overriding the filename.

```
zp c archive.zpaq /dir1/file1      (Warning: starts with "/")
zp x archive.zpaq                  (Error: bad filename)
zp e archive.zpaq                  (OK: extracts file1 to current directory)
zp x archive.zpaq newfile          (OK: extracts newfile to current directory)
zp x archive.zpaq /dir3/newfile    (OK: creates /dir3 if needed)
```

Also, the same rule applies to file names containing control characters, or longer than 511 characters, or that start with a drive letter like “C:” or that go up directories (contain *../*).

If this program is run in Linux or UNIX or compiled with g++ in Windows then it will interpret wildcards on the command line in the usual way. A *** matches any string and *?* matches any character.

```
zp c archive.zpaq *
```

will compress all files in the current directory to *archive.zpaq*. However, it will not recurse directories. You need to specify the files in each directory that you want to add.

The program does not save file timestamps or permissions like some other archivers do. Extracted files are dated from the time of extraction with default permissions. If you need these capabilities, then create a tar file and compress that instead.

The compression option 1, 2, or 3 means compress fast, medium, or best respectively. Better compression requires more time and memory. Decompression speed and memory are the same as for compression. Speed (T3200, 2.0 GHz) and memory usage are as follows. The following table shows comparison to zip -9 compression All modes compress better but slower than zip.

	Memory	Speed	Calgary corpus
	-----	-----	-----
1 (fast)	38 MB	0.7 sec/MB	807,214 bytes
2 (default)	111 MB	2.3 sec/MB	699,586 bytes
3 (small)	246 MB	6.4 sec/MB	644,545 bytes
zip -9	<1 MB	0.13 sec/MB	1,020,719 bytes

zp(1) uses compiled ZPAQL (generated by *zpaq oc*) to compress and extract in each of the 3 modes about twice as fast as using interpreted code. It automatically recognizes these configurations even if they are produced by other programs. The default compression is the same as the default produced by *zpaq*(1) and *zpipe*(1). If another program produces a different configuration, then this program will still correctly decompress it by interpreting the code, which is slower. Also, *zpaq*(1), *unzpaq*(1), and *zpipe*(1) can decompress archives produced by this program.

The program stores a filename, comment, and SHA-1 checksum for each file. Other programs may omit these, but this program will still be able to decompress them. This program follows the convention that if the name is omitted, then the contents should be appended to the previous file. If the first filename is omitted, then you must supply it on the command line during extraction. Each filename on the command line replaces one named file in the archive.

head2 Commands

a[N]

Append to archive.

Value N regulates the speed of compression using the specified digit: 1 (fast, less compression), 2 (medium, default), 3 (best, highest compression).

c[N]

Create archive.

Value N regulates the speed of compression using the specified digit: 1 (fast, less compression), 2 (medium, default), 3 (best, highest compression).

e[N]

Extract to current directory.

With N, extract only block N (1, 2, 3...), where 1 is the first block. Otherwise all blocks are extracted. The **I** command shows which files are in each block.

I

List contents of archive.

x[N]

Extract with full path names (files... overrides stored names).

With N, extract only block N (1, 2, 3...), where 1 is the first block. Otherwise all blocks are extracted. The **I** command shows which files are in each block.

OPTIONS

None.

EXAMPLES

Create

The archive name must end with *.zpaq*. All commands will add the extension automatically if you don't specify it.

To create an archive:

```
zp c3 archive.zpaq files ...
```

File names are stored in the archive as they appear on the command line. If you specify a path to a different directory, the path is stored, and created during extraction. The **e** command extracts to the current directory.

Append

To (a)ppend to an existing archive. If the archive does not exist then it is created as with the `c` command. The files are grouped into blocks (solid archive) for each command (see **I** command).

```
zp a3 archive.zpaq files ...
```

List

To list the contents of an archive. Files are listed in the same block order they were added:

```
zp l archive.zpaq
```

Extract

To extract the contents of the archive:

```
zp x archive.zpaq
```

To extract specific block (see **I** command output):

```
zp x1 archive.zpaq
```

Blocks are “solid” which means you cannot extract files within a block without extracting the earlier files. To extract first file in block under another name:

```
zp x1 archive.zpaq other-name
```

Program will not overwrite existing files during extraction unless you specify the filenames on the command line:

```
zp x archive.zpaq                (Error: file1 exists)
zp x archive.zpaq file1 file2    (Overwrite file1, file2)
```

ENVIRONMENT

None.

FILES

None.

STANDARDS

See `zpaq*.pdf` (ZPAQ Level 1 and later) in section AVAILABILITY. It is anticipated that future levels (ZPAQ-2, ZPAQ-3, etc.) will be backward compatible, such that newer levels can read archives produced by older programs.

AVAILABILITY

<<http://mattmahoney.net/dc>>

SEE ALSO

bzip2(1) *gzip*(1) *lzma*(1) *lzop*(1) *p7zip*(1) *rzip*(1) *unace*(1) *unrar*(1) *unzip*(1) *zip*(1) *zpaq*(1)

AUTHORS

Program was written by Matt Mahoney <matmahoney@yahoo.com>

This manual page was written by Jari Aalto <jari.aalto@cante.net>. Released under license GNU GPL version 3 or (at your option) any later version. For more information about license, visit <<http://www.gnu.org/copyleft/gpl.html>>.