IEEE P1003.2a Draft 8 – December 1991

Copyright © 1991 by the
Institute of Electrical and Electronics Engineers, Inc.
345 East 47th Street
New York, NY 10017, USA
All rights reserved as an unpublished work.

This is an unapproved and unpublished IEEE Standards Draft, subject to change. The publication, distribution, or copying of this draft, as well as all derivative works based on this draft, is expressly prohibited except as set forth below.

Permission is hereby granted for IEEE Standards Committee participants to reproduce this document for purposes of IEEE standardization activities only, and subject to the restrictions contained herein.

Permission is hereby also granted for member bodies and technical committees of ISO and IEC to reproduce this document for purposes of developing a national position, subject to the restrictions contained herein.

Permission is hereby also granted to the preceding entities to make limited copies of this document in an electronic form only for the stated activities.

The following restrictions apply to reproducing or transmitting the document in any form: 1) all copies or portions thereof must identify the document's IEEE project number and draft number, and must be accompanied by this entire notice in a prominent location; 2) no portion of this document may be redistributed in any modified or abridged form without the prior approval of the IEEE Standards Department.

Other entities seeking permission to reproduce this document, or any portion thereof, for standardization or other activities, must contact the IEEE Standards Department for the appropriate license.

Use of information contained in this unapproved draft is at your own risk.

IEEE Standards Department
Copyright and Permissions
445 Hoes Lane, P.O. Box 1331
Piscataway, NJ 08855-1331, USA
+1 (908) 562-3800
+1 (908) 562-1571 [FAX]

omitted `-t` because it seemed outside the UPE scope; it was not described in any of the Base Documents. However, a good case was made that hard-coding tab stops every eight columns was not suitable for the international community and broke historical precedents for some vendors in the FORTRAN community, so `-t` was restored in conjunction with the list of valid extension categories considered by the working group. Thus, `unexpand` is now the logical converse of `expand`, as one might expect.

One nonintuitive aspect of `unexpand` is its restriction to leading spaces when `-a` (and now, `-t`) are not specified. This is historical practice and there is no compelling reason to change it. Users who desire converting all spaces in the file can easily alias `unexpand` to use the `-a` or `-t 8` option.

5.33 `uudecode` — Decode a binary file

5.33.1 Synopsis

`uudecode` [*file*]

5.33.2 Description

The `uudecode` utility shall read a file, or standard input if no file is specified, that includes data created by the `uuencode` utility (see 5.34). The `uudecode` utility shall scan the input file, searching for data compatible with the format specified in 5.34.6.1 and attempt to create or overwrite the file described by the data. The pathname, file access permission bits, and contents for the file to be produced shall all be contained in that data. The mode bits of the created file shall be set from the file access permission bits contained in the data; i.e., other attributes of the mode, including the file mode creation mask (see `umask` in 4.67), shall not affect the file being produced.

If the pathname of the file to be produced exists, and the user does not have write permission on that file, `uudecode` shall terminate with an error. If the pathname of the file to be produced exists, and the user has write permission on that file, the existing file shall be overwritten.

If the input data was produced by `uuencode` on a system with a different number of bits per byte than on the target system, the results of `uudecode` are unspecified.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

5.33.3 Options

None.

5.33.4 Operands

The following operand shall be supported by the implementation:

file The pathname of a file containing the output of uuencode; see 5.34.6.1.

5.33.5 External Influences

5.33.5.1 Standard Input

See Input Files.

5.33.5.2 Input Files

The input files shall be files containing the output of uuencode; see 5.34.6.1.

5.33.5.3 Environment Variables

The following environment variables shall affect the execution of uudecode:

LANG This variable shall determine the locale to use for the locale categories when both **LC_ALL** and the corresponding environment variable (beginning with **LC_**) do not specify a locale. See 2.6.

LC_ALL This variable shall determine the locale to be used to override any values for locale categories specified by the settings of **LANG** or any environment variables beginning with **LC_**.

LC_CTYPE This variable shall determine the locale for the interpretation of sequences of bytes of text data as characters (e.g., single- versus multibyte characters in arguments and input files). 8 8

LC_MESSAGES This variable shall determine the language in which messages should be written.

5.33.5.4 Asynchronous Events

Default.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

5.33.6 External Effects

5.33.6.1 Standard Output

None.

5.33.6.2 Standard Error

Used only for diagnostic messages.

5.33.6.3 Output Files

The output file shall be in the same format as the file originally encoded by uuencode.

5.33.7 Extended Description

None.

5.33.8 Exit Status

The uudecode utility shall exit with one of the following values:

0 Successful completion.

>0 An error occurred.

5.33.9 Consequences of Errors

Default.

5.33.10 Rationale. *(This subclause is not a part of P1003.2a)*

Usage, Examples

The user who is invoking uudecode must have write permission on the specified file.

History of Decisions Made

Input files are not necessarily text files, as stated by an earlier draft. Although the uuencode output is a text file, that output could have been wrapped within another file or mail message that is not a text file.

The output of uuencode is essentially an encoded bit stream that is not cognizant of byte boundaries. It is possible that a 9-bit byte target machine can process input from an 8-bit source, if it is aware of the requirement, but the reverse is unlikely to be satisfying. Of course, the only data that is meaningful for such a

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

transfer between architectures is generally character data.

5.34 uuencode — Encode a binary file

5.34.1 Synopsis

`uuencode` [*file*] *decode_pathname*

5.34.2 Description

The `uuencode` utility shall write an encoded version of the named input file, or standard input if no *file* is specified, to standard output. The output shall be encoded using the algorithm described in 5.34.6.1, and shall include the file access permission bits (in `chmod` octal or symbolic notation; see 4.7) of the input file and the *decode_pathname* , for recreation of the file on another system that conforms to this standard.

5.34.3 Options

None.

5.34.4 Operands

The following operands shall be supported by the implementation:

decode_pathname
: The pathname of the file into which the `uudecode` utility (see 5.33) shall place the decoded file. If there are characters in *decode_pathname* that are not in the portable filename character set (see 2.2.2.111 in POSIX .1 {8}),the results are unspecified.

file
: A pathname of the file to be encoded.

5.34.5 External Influences

5.34.5.1 Standard Input

See Input Files.

5.34.5.2 Input Files

Input files can be files of any type.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.