Copyright (c) 1993-2009 Ingo Chao, Thomas Bernard, Xavier Roy, Glenn Randers-Pehrson
This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.
Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:
1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.
2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.
3. This notice may not be removed or altered from any source distribution.
Copyright (c) 1993-2009 Ingo Chao, Thomas Bernard, Xavier Roy, Glenn Randers-Pehrson
This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.
Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:
1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.
2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.
3. This notice may not be removed or altered from any source distribution.

```
/* zlib.h -- interface of the 'zlib' general purpose compression library
  version 1.2.3, July 18, 2005

  Copyright (C) 1995-2005 Jean-loup Gailly and Mark Adler

  This software is provided 'as-is', without any express or implied
  warranty.  In no event will the authors be held liable for any damages
  arising from the use of this software.

  Permission is granted to anyone to use this software for any purpose,
  including commercial applications, and to alter it and redistribute it
  freely, subject to the following restrictions:

  1. The origin of this software must not be misrepresented; you must not
     claim that you wrote the original software. If you use this software
     in a product, an acknowledgment in the product documentation would
     be appreciated but is not required.
  2. Altered source versions must be plainly marked as such, and must not
     be misrepresented as being the original software.
  3. This notice may not be removed or altered from any source
     distribution.

  Jean-loup Gailly        Mark Adler
  jloup@gzip.org          madler@alumni.caltech.edu

  The data format used by the zlib library is described by RFCs (Request for
  Comments) 1950 to 1952 in the files http://ftp.uu.net/pub/doc/rfc/rfc1950.txt
  (zlib format), rfc1951.txt (deflate format) and rfc1952.txt (gzip format).
*/

#ifndef ZLIB_H
#define ZLIB_H

#define ZLIB_VERSION "1.2.3"
#define ZLIB_VERNUM 0x1230
#define ZLIB_VER_MAJOR 1
#define ZLIB_VER_MINOR 2
#define ZLIB_VER_REVISION 3
#define ZLIB_VER_SUBREVISION 0

#define zlib_version zlibVersion()
/* Tuple of library version.
   The first two digits are the major version number.
   The last two digits are the minor version number.
   The last four digits are the revision and subrevision.
   For example, 1.2.3.0 is version 1.2, revision 3, subrevision 0
*/

#define Z_NO_FLUSH      0
#define Z_PARTIAL_FLUSH 1
#define Z_SYNC_FLUSH    2
#define Z_FULL_FLUSH    3
#define Z_FINISH        4
#define Z_BLOCK         5
#define Z_TREES         6
/* Allowable flush values; see deflate() and inflate() below
   for details
*/

#define Z_OK            0
#define Z_STREAM_END    1
#define Z_NEED_DICT     2
#define Z_ERRNO        (-1)
#define Z_STREAM_ERROR (-2)
#define Z_DATA_ERROR   (-3)
#define Z_MEM_ERROR    (-4)
#define Z_BUF_ERROR    (-5)
#define Z_VERSION_ERROR (-6)
/* Return codes for the compression/decompression functions. Negative
   values are errors, positive values are used for special but
   normal events.
*/

#define Z_NO_COMPRESSION         0
#define Z_BEST_SPEED             1
#define Z_BEST_COMPRESSION       9
#define Z_DEFAULT_COMPRESSION  (-1)
/* compression levels
*/

#define Z_FILTERED            1
#define Z_HUFFMAN_ONLY        2
#define Z_RLE                 3
#define Z_FIXED               4
#define Z_DEFAULT_STRATEGY    0
/* compression strategy; see deflate() for details
*/

#define Z_BINARY  0
#define Z_ASCII   1
#define Z_UNKNOWN 2
/* Possible values of the data_type field (though see inflate())
*/

#define Z_DEFLATED   8
/* The deflate compression method
   (also known as Huffman)
*/

#define Z_NULL  0  /* for initializing zalloc, zfree, opaque
*/

#define zalloc Z_NULL
#define zfree  Z_NULL
#define opaque Z_NULL

typedef voidpf (*alloc_func OF((voidpf opaque, uInt items, uInt size));
typedef void   (*free_func OF((voidpf opaque, voidpf address));

struct internal_state;
typedef struct z_stream_s {
    Bytef    *next_in;  /* next input byte
    uInt     avail_in;  /* number of bytes available at next_in
    uLong    total_in;  /* total nb of input bytes read so far
    Bytef    *next_out; /* next output byte will go here
    uInt     avail_out; /* remaining free space at next_out
    uLong    total_out; /* total nb of bytes output so far
    char     *msg;      /* last error message, NULL if no error
    struct internal_state FAR *state; /* not visible by applications
    alloc_func zalloc;  /* used to allocate the internal state
    free_func  zfree;   /* used to free the internal state
    voidpf     opaque;  /* private data object passed to zalloc and zfree
    int        data_type;  /* best guess about the data type: binary or text
    uLong      adler;      /* adler32 value of the uncompressed data
    uLong      reserved;   /* reserved for future use
} z_stream;

typedef z_stream FAR *z_streamp;

/* The application must update next_in and avail_in when data is
   available in the input buffer. When the compressor or decompressor
   produces output, it updates next_out and avail_out accordingly.
   The pointers to next_in, next_out, and the corresponding
   lengths may be set to NULL on initialization.
*/

/* gzip header information passed to and from zlib routines.
   See RFC 1952 for more detailed definitions.
*/
typedef struct gzip_header {
    int  text;  /* true if compressed data believed to be text
    uLong time;
    int  xflags;
    int  os;
    Bytef *extra;  /* pointer to extra field or Z_NULL if none
    uInt  extra_max;
    uInt  extra_len;
    Bytef *name;   /* pointer to zero-terminated file name or Z_NULL
    uInt  name_max;
    uInt  name_len;
    uLong hcrc;    /* crc32 of header if header_crc is 1
    Bytef *comment;/* pointer to zero-terminated comment or Z_NULL
    uInt  comment_max;
    uInt  comment_len;
    int  done;  /* true when done reading gzip header (not used
                   when writing or inflating)
} gzip_header;

typedef gzip_header FAR *gzip_headerp;

ZEXTERN const char * ZEXPORT zlibVersion OF((void));
ZEXTERN int ZEXPORT deflateInit OF((z_streamp strm, int level));
ZEXTERN int ZEXPORT deflate OF((z_streamp strm, int flush));
ZEXTERN int ZEXPORT deflateEnd OF((z_streamp strm));
ZEXTERN int ZEXPORT inflateInit OF((z_streamp strm));
ZEXTERN int ZEXPORT inflate OF((z_streamp strm, int flush));
ZEXTERN int ZEXPORT inflateEnd OF((z_streamp strm));
ZEXTERN int ZEXPORT deflateInit2 OF((z_streamp strm, int level, int method,
                                     int windowBits, int memLevel,
                                     int strategy));
ZEXTERN int ZEXPORT deflateSetDictionary OF((z_streamp strm,
                                             const Bytef *dictionary,
                                             uInt dictLength));
ZEXTERN int ZEXPORT deflateCopy OF((z_streamp dest,
                                    z_streamp source));
ZEXTERN int ZEXPORT deflateReset OF((z_streamp strm));
ZEXTERN int ZEXPORT deflateParams OF((z_streamp strm,
                                      int level,
                                      int strategy));
ZEXTERN int ZEXPORT deflateBound OF((z_streamp strm,
                                     uLong sourceLen));
ZEXTERN int ZEXPORT deflatePrime OF((z_streamp strm,
                                     int bits,
                                     int value));
ZEXTERN int ZEXPORT deflateSetHeader OF((z_streamp strm,
                                         gzip_headerp head));
ZEXTERN int ZEXPORT inflateBackInit OF((z_streamp strm,
                                        int windowBits,
                                        char FAR *window));
ZEXTERN int ZEXPORT inflateBack OF((z_streamp strm,
                                    in_func in,
                                    voidpf FAR *in_desc,
                                    out_func out,
                                    voidpf FAR *out_desc));
ZEXTERN int ZEXPORT inflateBackEnd OF((z_streamp strm));
ZEXTERN uLong ZEXPORT zlibCompileFlags OF((void));
/* Return flags indicating compile-time options.
   Type is a bit vector, bit 0 is set if sizeof(uInt) = 2,
   bit 1 is set if sizeof(uLong) = 2, bit 2 is set if sizeof(voidpf) = 2,
   bit 3 is set if sizeof(z_off_t) = 2, bit 4 is set if neither uInt nor
   uLong is 32 bits, bit 5 is set if the compiler supports long long,
   bit 6 is set if the compiler supports the size_t type, bit 7 is set
   if the compiler supports the ptrdiff_t type, bit 8 is set if
   the compiler supports the ptrdiff_t type, bit 9 is set if
   the compiler supports the ptrdiff_t type, bit 10 is set if
   the compiler supports the ptrdiff_t type, bit 11 is set if
   the compiler supports the ptrdiff_t type, bit 12 is set if
   the compiler supports the ptrdiff_t type, bit 13 is set if
   the compiler supports the ptrdiff_t type, bit 14 is set if
   the compiler supports the ptrdiff_t type, bit 15 is set if
   the compiler supports the ptrdiff_t type, bit 16 is set if
   the compiler supports the ptrdiff_t type, bit 17 is set if
   the compiler supports the ptrdiff_t type, bit 18 is set if
   the compiler supports the ptrdiff_t type, bit 19 is set if
   the compiler supports the ptrdiff_t type, bit 20 is set if
   the compiler supports the ptrdiff_t type, bit 21 is set if
   the compiler supports the ptrdiff_t type, bit 22 is set if
   the compiler supports the ptrdiff_t type, bit 23 is set if
   the compiler supports the ptrdiff_t type, bit 24 is set if
   the compiler supports the ptrdiff_t type, bit 25 is set if
   the compiler supports the ptrdiff_t type, bit 26 is set if
   the compiler supports the ptrdiff_t type, bit 27 is set if
   the compiler supports the ptrdiff_t type, bit 28 is set if
   the compiler supports the ptrdiff_t type, bit 29 is set if
   the compiler supports the ptrdiff_t type, bit 30 is set if
   the compiler supports the ptrdiff_t type, bit 31 is set if
   the compiler supports the ptrdiff_t type, bit 32 is set if
   the compiler supports the ptrdiff_t type, bit 33 is set if
   the compiler supports the ptrdiff_t type, bit 34 is set if
   the compiler supports the ptrdiff_t type, bit 35 is set if
   the compiler supports the ptrdiff_t type, bit 36 is set if
   the compiler supports the ptrdiff_t type, bit 37 is set if
   the compiler supports the ptrdiff_t type, bit 38 is set if
   the compiler supports the ptrdiff_t type, bit 39 is set if
   the compiler supports the ptrdiff_t type, bit 40 is set if
   the compiler supports the ptrdiff_t type, bit 41 is set if
   the compiler supports the ptrdiff_t type, bit 42 is set if
   the compiler supports the ptrdiff_t type, bit 43 is set if
   the compiler supports the ptrdiff_t type, bit 44 is set if
   the compiler supports the ptrdiff_t type, bit 45 is set if
   the compiler supports the ptrdiff_t type, bit 46 is set if
   the compiler supports the ptrdiff_t type, bit 47 is set if
   the compiler supports the ptrdiff_t type, bit 48 is set if
   the compiler supports the ptrdiff_t type, bit 49 is set if
   the compiler supports the ptrdiff_t type, bit 50 is set if
   the compiler supports the ptrdiff_t type, bit 51 is set if
   the compiler supports the ptrdiff_t type, bit 52 is set if
   the compiler supports the ptrdiff_t type, bit 53 is set if
   the compiler supports the ptrdiff_t type, bit 54 is set if
   the compiler supports the ptrdiff_t type, bit 55 is set if
   the compiler supports the ptrdiff_t type, bit 56 is set if
   the compiler supports the ptrdiff_t type, bit 57 is set if
   the compiler supports the ptrdiff_t type, bit 58 is set if
   the compiler supports the ptrdiff_t type, bit 59 is set if
   the compiler supports the ptrdiff_t type, bit 60 is set if
   the compiler supports the ptrdiff_t type, bit 61 is set if
   the compiler supports the ptrdiff_t type, bit 62 is set if
   the compiler supports the ptrdiff_t type, bit 63 is set if
   the compiler supports the ptrdiff_t type
*/

/* Utility functions.
   These functions are not compiled by default.
   To use them, define the preprocessor symbol ZLIB_INTERNAL.
*/
ZEXTERN uLong ZEXPORT adler32 OF((uLong adler, const Bytef *buf,
                                  uInt len));
ZEXTERN uLong ZEXPORT adler32_combine OF((uLong adler1,
                                           uLong adler2,
                                           z_off_t len));
ZEXTERN uLong ZEXPORT crc32 OF((uLong crc, const Bytef *buf,
                                uInt len));
ZEXTERN uLong ZEXPORT crc32_combine OF((uLong crc1,
                                         uLong crc2,
                                         z_off_t len));

/* Compression stream structure.
   This structure is used by the compression functions.
   It contains the compressed data and the uncompressed data.
   The pointers to next_in and next_out are updated by the functions.
   The pointers to next_in and next_out are updated by the functions.
   The pointers to next_in and next_out are updated by the functions
```

```
        -7 '-7 REFUSED'
        -8 '-8 DK'
        -9 '-9 NOT ASCERTAINED'
        1 '1 NO DIFFICULTY'
        2 '2 SOME DIFFICULTY'
        3 '3 A LOT OF DIFFICULTY'
        4 '4 COMPLETELY UNABLE TO DO IT'
        5 '5 COMPLETELY UNABLE TO WALK' /
    STPRG13
        -1 '-1 INAPPLICABLE'
        1 '1 YES'
        2 '2 NO' /
    TRINW13X
        -1 '-1 INAPPLICABLE'
        1 '1 YES'
        2 '2 NO' /
    UNABLE13
        -1 '-1 INAPPLICABLE'
        -7 '-7 REFUSED'
        -8 '-8 DK'
        -9 '-9 NOT ASCERTAINED'
        1 '1 YES'
        2 '2 NO' /
    WGTRU13
        -1 '-1 INAPPLICABLE'
        0 '0.000000 WEIGHT' /
    WGTSP13
        -1 '-1 INAPPLICABLE'
        0 '0.000000 WEIGHT' /
    WLKDIF13
        -1 '-1 INAPPLICABLE'
        -7 '-7 REFUSED'
        -8 '-8 DK'
        -9 '-9 NOT ASCERTAINED'
        1 '1 NO DIFFICULTY'
        2 '2 SOME DIFFICULTY'
        3 '3 A LOT OF DIFFICULTY'
        4 '4 UNABLE TO DO' /
    WLKLIM13
        -1 '-1 INAPPLICABLE'
        -7 '-7 REFUSED'
        -8 '-8 DK'
        -9 '-9 NOT ASCERTAINED'
        1 '1 YES'
        2 '2 NO' /
    WRKLIM13
        -1 '-1 INAPPLICABLE'
        -7 '-7 REFUSED'
        -8 '-8 DK'
        -9 '-9 NOT ASCERTAINED'
        1 '1 YES'
        2 '2 NO' /
    YRSINUS
        -1 '-1 INAPPLICABLE'
        -7 '-7 REFUSED'
        -8 '-8 DK'
        -9 '-9 NOT ASCERTAINED'
        1 '1 LESS THAN 1 YEAR'
        2 '2 1 YR., LESS THAN 5 YRS.'
        3 '3 5 YRS., LESS THAN 10 YRS.'
        4 '4 10 YRS., LESS THAN 15 YRS.'
        5 '5 15 YEARS OR MORE' .
EXECUTE.
```