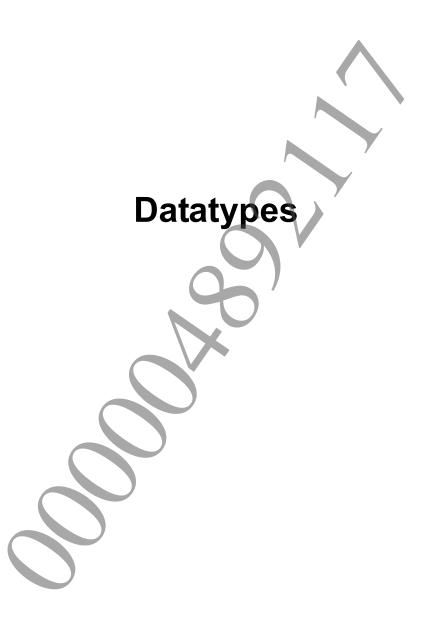


© 2011 Sony Computer Entertainment Inc. All Rights Reserved. SCE Confidential

# **Table of Contents**

Datatypes	3
SceGzipHeader	4
SceZipHeaderPK0304	6
SceZipFooterPK0708	7
SceDeflateDecompressPartialInputCallback	8
SceDeflatePartialInputParam	9
Expansion Functions	10
sceDeflateDecompress	
sceDeflateDecompressPartial	12
sceGzipDecompress	13
sceZlibDecompress	14
GZIP Header Parsing Functions	15
sceGzipGetComment	
sceGzipGetCompressedData	17
sceGzipGetInfo	18
sceGzipGetName	19
sceGzipIsValid	20
sceGzipGetNamesceGzipIsValidZLIB Header Parsing Functions	21
sceZlibGetCompressedData	22
sceZlibGetInfo	23
sceZlibIsValid	24
ZIP Header Parsing Function	25
sceZipGetInfo	
CRC32 Functions	
sceGzipCrc32	
Adler-32 Checksum Functions	
sceZlibAdler32	
Constants	
List of Error Codes	32



# SceGzipHeader

#### **GZIP-format header**

#### **Definition**

```
#include <libdeflt.h>
typedef struct {
    unsigned char id1;
    unsigned char id2;
    unsigned char cm;
    unsigned char flg;
    unsigned int uiMtime;
    unsigned char xlf;
    unsigned char os;
} SceGzipHeader;
```

#### **Members**

```
id1
          Magic number (fixed to 0x1F)
id2
          Magic number (fixed to 0x8B)
          Indicates the compression method. Only 0x08=DEFLATE is supported by libdeflt.
сm
flg
          Flags
          Bits 7 - 5: Reserved
          Bit 4: FCOMMENT
          Bit 3: FNAME
          Bit 2: FEXTRA
          Bit 1: FHCRC
          Bit 0: FTEXT
uiMtime
          File Modified Datetime (GMT)
x1f
          Extra flags
          0x02 = Maximum compression rate
          0x04 = Prioritize speed
          OS identification code
os
          0x00 = FAT
          0x01 = Amiga
          0x02 = VMS
          0x03 = Unix
          0x04 = VM/CMS
          0x05 = Atari TOS
          0x06 = HPFS (OS/2, NT)
          0x07 = Macintosh
          0x08 = Z-System
          0x09 = CP/M
          0x0A = TOPS-20
          0x0B = NTFS (NT)
          0x0C = QDOS
```

#### Description

This structure represents the header of a GZIP-format compressed file.

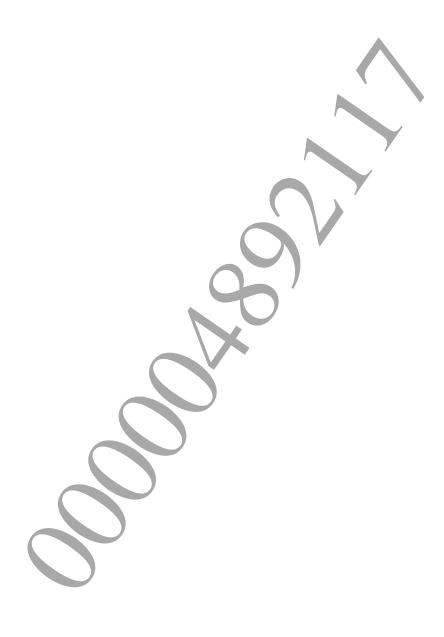
0x0D = Acorn RISCOS 0xFF = Unknown

When GZIP header parsing functions are used, the value of each field, the start address, etc., can be obtained.

**©SCEI** 

# See Also

sceGzipGetInfo(),sceGzipGetName(),sceGzipGetComment(), sceGzipGetCompressedData()



# SceZipHeaderPK0304

#### ZIP header attached to each file

#### **Definition**

```
#include typedef struct {
        unsigned int signature;
        unsigned short version;
        unsigned short option;
        unsigned short cm;
        unsigned short filetime;
        unsigned short filedate;
        unsigned int crc32;
        unsigned int uncompsize;
        unsigned int uncompsize;
        unsigned short fnamelen;
        unsigned short extralen;
        char filename[1];
} __attribute__((packed)) SceZipHeaderPK0304;
```

#### **Members**

signature Magic number (fixed to 0x04034b50) version Version option Option Indicates the compression method. Only 0x08=DEFLATE is supported by libdeflt. cmfiletime File update time (MS-DOS format) File update date (MS-DOS format) filedate crc32 **CRC** compsize Compressed size uncompsize Uncompressed size fnamelen Filename length extralen Extra field length filename Filename (variable length, fnamelen bytes with no terminating character)

#### **Description**

This structure represents the header of a ZIP-format compressed file. Multiple files can be collected together (archived) in a ZIP file, and a header with this format is attached to each individual archived file.

When ZIP header parsing functions are used, this header is interpreted and the value of each field, the start address, etc., can be obtained.

## See Also

```
sceZipGetInfo()
```

# SceZipFooterPK0708

# ZIP footer attached to each file

#### **Definition**

```
#include <libdeflt.h>
typedef struct {
    unsigned int signature;
    unsigned int crc32;
    unsigned int compsize;
    unsigned int uncompsize;
}
__attribute__((packed)) SceZipFooterPK0708;
```

#### **Members**

signature Magic number (fixed to 0x08074b50)

crc32 CRC

compsize Compressed size uncompsize Uncompressed size

# **Description**

This structure represents the footer of a ZIP-format compressed file. When the header attached to each file does not contain information related to the CRC and file size, a footer with this format is assigned to store that information.

When ZIP header parsing functions are used, this footer is interpreted and the appropriate CRC32 value can be obtained.

# See Also

sceZipGetInfo()



# SceDeflateDecompressPartialInputCallback

Type of callback function used when performing partial input of data in DEFLATE format

#### **Definition**

#### **Members**

param Pointer to structure for exchanges of information such as the data position.

The pointer given when the  ${\tt sceDeflateDecompressPartial}$  () function is called

is passed as is.

outputsize Size of data expanded at the time callback is called.

## **Description**

This is the callback function that is called if input data falls short along the way when DEFLATE format data is expanded by breaking the input data into smaller segments (i.e. allowing data expansion using smaller data sizes).

When this function is called, after preparing the data with an appropriate method based on <code>param->cookie</code>, etc., update <code>param->pBufEnd</code> with the pointer value pointing to the address of the new input data end + 1, and return the pointer to the new input data beginning as the return value.

Both the input data start address and the end + 1 address must be multiples of 4. If this condition is not met, the sceDeflateDecompressPartial() function returns

SCE DEFLATE ERROR INVALID POINTER.

The expanded data size is given to *outputsize*. By checking this value, it is possible to perform some processing or other when data expansion has been done to a certain size, or else stop data expansion.

#### See Also

sceDeflateDecompressPartial(

# SceDeflatePartialInputParam

Structure holding data used for partial input of data in DEFLATE format

#### **Definition**

```
#include <libdeflt.h>
struct SceDeflatePartialInputParam;

typedef struct SceDeflatePartialInputParam {
         unsigned int size;
         const void *pBufEnd;
         void *cookie;
         SceDeflateDecompressPartialInputCallback *callback;
} SceDeflatePartialInputParam;
```

#### **Members**

size Size of this structure

pBufEnd Input data buffer end address + 1

cookie Arbitrary value that can be used freely by callback function

callback Callback function

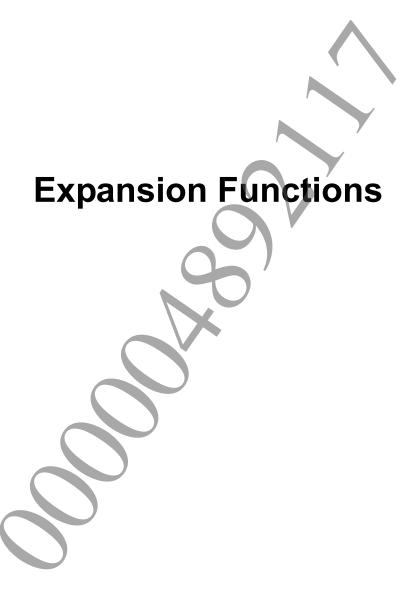
#### **Description**

By using the sceDeflateDecompressPartial () function, DEFLATE format data can be expanded by breaking the input data into smaller segments (i.e. allowing data expansion using smaller buffer sizes). At this time, this structure holds the data to be exchanged, such as the addition method for partial input data and the addition result.

### See Also

sceDeflateDecompressPartial()





# sceDeflateDecompress

# Expand DEFLATE-compressed data

#### **Definition**

# **Calling Conditions**

Multithread safe

### **Arguments**

pDst uiBufSize pSrcDeflate ppNext Specifies the buffer address which is to receive the expanded data.

Specifies the size of the buffer which is to receive the expanded data.

Specifies the start address of the DEFLATE-compressed compressed data.

Specifies a pointer to the void \* variable which returns the address immediately

following the processed compressed data after expansion.

## Return value

Value	Result
0 or higher	Size of the expanded data
SCE_DEFLATE_ERROR_INVALID_FORMAT	Invalid format
SCE DEFLATE ERROR INVALID SIZE	Buffer overflowed during expansion

#### **Description**

Expands DEFLATE-compressed data.

This function takes the start address of the DEFLATE-compressed data. To expand data with a GZIP header, retrieve the start address of the DEFLATE-compressed data using the

 $\verb|sceGzipGetCompressedData|()|, or use the \verb|sceGzipDecompress|()|.$ 

If expansion is completed normally, the address immediately following the compressed data which was processed is assigned to the void \* variable which was specified by the <code>ppNext</code> argument. For the GZIP format, CRC32 and ISIZE are stored, and for the ZLIB format, ADLER32 is stored.

#### See Also

sceGzipDecompress(), sceGzipGetCompressedData()

# sceDeflateDecompressPartial

# Expansion of divided DEFLATE compressed data

#### **Definition**

## **Calling Conditions**

Multithread safe

## **Arguments**

pDst Specifies the buffer address to receive the expanded data

uiBufSize Specifies the size of the buffer to receive the expanded data

pSrcDeflate Specifies the start address of the DEFLATE format compressed data

ppNext Specifies the pointer to the void\* variable that returns the address immediately

following after the processed compressed data after expansion

cbInfo Specifies the pointer to the structure holding information such as the callback function

for adding data when insufficient input data is detected.

#### **Return Values**

Value	Result
0 or higher	Size of the expanded data
SCE_DEFLATE_ERROR_INVALID_FORM	Invalid format
SCE_DEFLATE_ERROR_INVALID_SIZE	Buffer overflowed during expansion
SCE_DEFLATE_ERROR_INVALID_CBINE	Invalid callback structure specification

#### Description

Expands DEFLATE compressed data.

This function is basically the same as the sceDeflateDecompress() function, except that it allows breaking down the input data into smaller segments. When the input data is insufficient, the <code>cbInfo.callback</code> function is called. Since this function can be freely set in the user application, any processing (such as input from a file or network) can be done to add insufficient data.

The expansion speed is slightly slower than when using the sceDeflateDecompress() function, because insufficient input data is additionally checked for.

#### See Also

sceGzipDecompress(), sceGzipGetCompressedData()

# sceGzipDecompress

# **Expand GZIP-format data**

#### **Definition**

# **Calling Conditions**

Multithread safe

# **Arguments**

pDst Specifies the buffer address which is to receive the expanded data.
 uiBufSize Specifies the size of the buffer which is to receive the expanded data.
 Specifies the start address of the GZIP-format compressed data.
 Specifies a pointer to the unsigned int variable to receive the CRC32 value.

#### Return value

Value	Result
0 or higher	Size of the expanded data
SCE_DEFLATE_ERROR_INVALID_FORMAT	Invalid format
SCE_DEFLATE_ERROR_INVALID_SIZE	Buffer overflowed during expansion
SCE DEFLATE ERROR NOT SUPPORTED	A compression method other than DEFLATE was used

#### **Description**

Expands DEFLATE-compressed data.

This function takes the start address of the data with a GZIP header.

## See Also

sceGzipCrc32(

# sceZlibDecompress

# Expand ZLIB-format data

#### **Definition**

# **Calling Conditions**

Multithread safe

# **Arguments**

pDst Specifies the buffer address which is to receive the expanded data. Specifies the size of the buffer which is to receive the expanded data. Specifies the start address of the ZLIB-format compressed data. puiAdler32 Specifies a pointer to the unsigned int variable which returns the

2 Specifies a pointer to the unsigned int variable which returns the Adler-32 checksum value stored at the address immediately following the processed

compressed data after expansion.

#### Return value

Value	Result
0 or higher	Size of the expanded data
SCE_DEFLATE_ERROR_INVALID_FORMAT	Invalid format
SCE_DEFLATE_ERROR_INVALID_SIZE	Buffer overflowed during expansion

#### **Description**

Expands DEFLATE-compressed data.

This function takes the start address of the ZLIB-format compressed data.

If expansion is completed normally, the Adler-32 checksum value stored at the address immediately following the compressed data which was processed is assigned to the unsigned int variable which was specified by the <code>puiAdler32</code> argument.

#### See Also

sceGzipDecompress(), sceGzipGetCompressedData(), sceZlibAdler32()



# sceGzipGetComment

# Get comment string

#### **Definition**

# **Calling Conditions**

Multithread safe

# **Arguments**

pSrcGzip Specifies the start address of the GZIP-format compressed data.

#### Return value

Value	Result		7
Non-NULL	Start address of the comment string		
NULL	A comment string is not stored in the G	ZIP ŀ	eader.
	Or, the data specified by pSrcGzip is r	ot a (	GZIP header.

# **Description**

Gets the address in which the comment string is stored in a GZIP header. If it is not GZIP-format data, or if a comment string is not contained in the header, NULL is returned.

This function is a wrapper function for the sceGzipGetInfo().

# See Also

sceGzipGetInfo()



# sceGzipGetCompressedData

Gets the start address of DEFLATE-compressed data

#### **Definition**

# **Calling Conditions**

Multithread safe

# **Arguments**

pSrcGzip Specifies the start address of the GZIP-format compressed data.

#### Return value

Value	Result
Non-NULL	Start address of DEFLATE-compressed data
NULL	The data specified by <i>pSrcGzip</i> is not in GZIP format.

# **Description**

Gets the start address of DEFLATE-compressed data from a GZIP header. If data is not in GZIP format, NULL is returned.

This function is a wrapper function for the sceGzipGetInfo().

## See Also

sceGzipGetInfo()

# sceGzipGetInfo

#### Get elements from GZIP header

### **Definition**

# **Calling Conditions**

Multithread safe

# **Arguments**

Specifies the start address of the GZIP-format compressed data. pSrcGzip ppvExtra Specifies a pointer to the void \* variable which receives the extra field start address. Specifies a pointer to the const char \* variable which receives the filename field ppszName start address. ppszComment Specifies a pointer to the const \* variable which receives the comment field char start address. Specifies a pointer to the unsigned short variable to receive the CRC16 value. pusCrc Specifies a pointer to the const void \* variable which receives the compressed data ppvData start address.

# Return value

Value		Result
0		Successful completion
SCE_DEFLATE_ERROR_I	NVALID_FORMAT	The GZIP header is invalid

#### **Description**

Gets the addresses of the elements from a GZIP file header. Only CRC16 is fixed-size data, so this element is retrieved directly.

Some of the elements may not be mandatory in GZIP headers. For elements which are not present in the file, NULL is returned for the address.

In the case that some elements are not necessarily retrieved, the retrieval processing can be omitted by specifying NULL to the corresponding pointer arguments which receive the results.

# Document serial number: 000004892117

# sceGzipGetName

# Get filename

### **Definition**

# **Calling Conditions**

Multithread safe

# **Arguments**

pSrcGzip Specifies the start address of the GZIP-format compressed data.

#### Return value

Value	Result		Y
Non-NULL	Start address of the filename		
NULL	A filename is not stored in the GZIP header.		
	Or, the data specified by pSrcGzip is	s not a	GZIP header.

# **Description**

Gets the address in which the filename is stored in a GZIP header. If it is not GZIP-format data, or if a filename is not contained in the header, NULL is returned.

This function is a wrapper function for the sceGzipGetInfo().

# See Also

sceGzipGetInfo()



# sceGzipIsValid

Check the magic number of a GZIP header

### **Definition**

# **Calling Conditions**

Multithread safe

# **Arguments**

pSrcGzip Specifies the start address of the GZIP-format compressed data.

#### Return value

Value	Result
1	Magic number found in GZIP header
0	Magic number does not match

## **Description**

Checks the magic number at the beginning of a header, and determines whether or not the data is in GZIP format. The first two bytes of the header of GZIP-format data constitute a magic number, and are 0x1F and 0x8B. sceGzipIsValid() checks only the first two bytes.





# sceZlibGetCompressedData

Get the start address of DEFLATE-compressed data

#### **Definition**

# **Calling Conditions**

Multithread safe

# **Arguments**

pSrcZlib Specifies the start address of the ZLIB-format compressed data.

#### Return value

Value	Result	
Non-NULL	Start address of DEFLATE-compressed data	
NULL	The data specified by pSrcZlib is not in ZLIB format	t.

# **Description**

Gets the start address of DEFLATE-compressed data from a ZLIB-format header. If data is not in ZLIB format, NULL is returned.

This function is a wrapper function for the sceZlibGetInfo().

## See Also

sceZlibGetInfo()

# sceZlibGetInfo

#### Get elements from ZLIB header

#### **Definition**

## **Calling Conditions**

Multithread safe

## **Arguments**

pSrcZlib Specifies the start address of the ZLIB-format compressed data.

pbCmf Specifies a pointer to the unsigned char variable which receives the compression

method and flags.

Specifies a pointer to the unsigned char variable which receives the flags for

compression level etc.

puiDictId Specifies a pointer to the unsigned int variable which receives the dictionary ID,

when a defined dictionary is used.

ppvData Specifies a pointer to the const void \* variable which receives the compressed data

start address.

#### Return value

Value		Result
0		Successful completion
SCE_DEFLATE_ERROR_	INVALID FORMAT	The ZLIB header is invalid

#### Description

Gets the elements from ZLIB-format data. For the meaning of the respective elements, refer to RFC 1950.

In the case that some elements are not necessarily retrieved, the retrieval processing can be omitted by specifying NULL to the corresponding pointer arguments which receive the results.

# sceZliblsValid

Check the magic number of a ZLIB header

#### **Definition**

# **Calling Conditions**

Multithread safe

# **Arguments**

pSrcZlib Specifies the start address of the ZLIB-format compressed data.

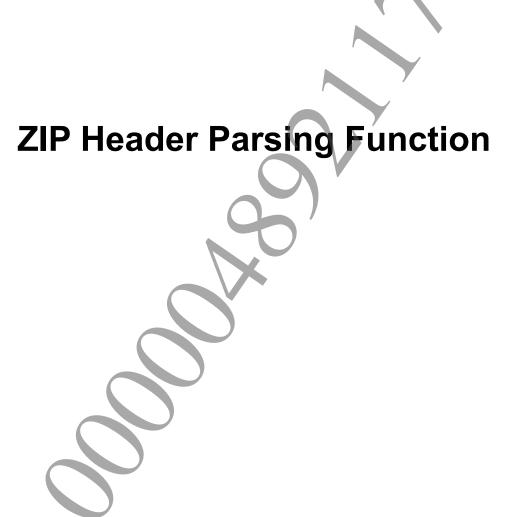
#### Return value

Value	Result
1	Magic number found in ZLIB header
0	Magic number does not match

## **Description**

Checks the magic number at the beginning of a ZLIB-format header, and determines whether or not the data is in ZLIB format. When the first two bytes of a ZLIB-format header constitute a big-endian 16-bit value, it will be an integral multiple of 31, sceZlibIsValid() checks only the first two bytes.





# sceZipGetInfo

Get elements from ZIP header

#### **Definition**

# **Calling Conditions**

Multithread safe

# **Arguments**

pSrc Specifies the start address of the ZIP-format compressed data.

ppvExtra Specifies a pointer to the void \* variable which receives the extra field start address.

puiCrc Specifies a pointer to the unsigned int variable to receive the CRC32 value.

ppvData Specifies a pointer to the const void \* variable which receives the compressed data

start address.

#### Return value

Value Result	Value Result
0	Successful completion
SCE DEFLATE ERROR INV	ALID FORMAT The ZIP header is invalid

### **Description**

Gets information from a ZIP-format archive related to individual files within the archive.

DEFLATE-format compressed data from the stored address, or uncompressed original data that is copied directly is placed at <code>ppvData</code>. If the data is compressed, it can be expanded by passing this address to <code>sceDeflateDecompress()</code>.

To determine if the data is compressed, check the value of the *cm* member of the SceZipHeaderPK0304 structure.



# Document serial number: 000004892117

# sceGzipCrc32

# Calculate CRC32 checksum

### **Definition**

# **Calling Conditions**

Multithread safe

# **Arguments**

uiCrc Specifies the initial value. Normally set to 0.
 pSrc Specifies the start address of the data for which to perform the CRC32 calculation.
 uiSize Specifies the size of the data for which to perform the CRC32 calculation.

## Return value

Returns a CRC32.

# **Description**

Calculates checksum digits using the CRC32 algorithm. Refer to RFC 1952.

# See Also

sceGzipDecompress()



# Document serial number: 000004892117

# sceZlibAdler32

# Calculate Adler-32 checksum

#### **Definition**

# **Calling Conditions**

Multithread safe

# **Arguments**

uiAdler Specifies the initial value. Normally set to 1.
 pSrc Specifies the start address of the data for which to perform the checksum calculation.
 uiSize Specifies the size of the data for which to perform the checksum calculation.

#### Return value

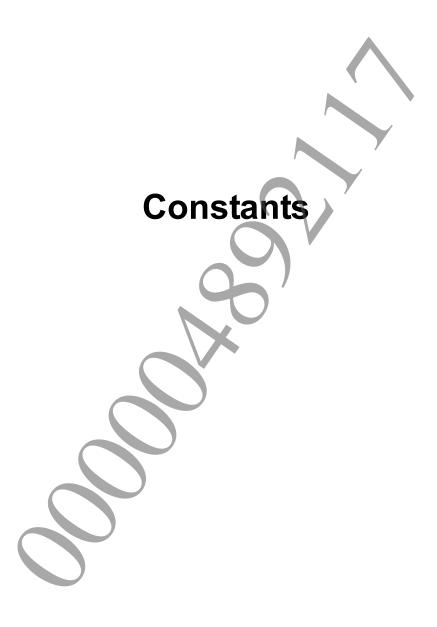
Returns an Adler-32-format checksum.

# **Description**

Calculates a checksum using the Adler-32 algorithm. Refer to RFC 1950.

# See Also

sceZlibDecompress()



# **List of Error Codes**

# libdeflt error codes

# Definition

Value	Result
0 or higher	Size of the expanded data
SCE_DEFLATE_ERROR_INVALID_FORMAT	Invalid format
SCE_DEFLATE_ERROR_INVALID_SIZE	Buffer overflowed during expansion
SCE_DEFLATE_ERROR_NOT_SUPPORTED	A compression method other than DEFLATE was used
SCE_DEFLATE_ERROR_INVALID_CBINFO	Invalid callback structure specification
SCE DEFLATE ERROR INVALID POINTER	Invalid pointer

