libatrac Reference

PlayStation®Vita Programmer Tool Runtime Library Release 3.50

Table of Contents

Datatypes

SceAtracDecoderGroup

Decoder group structure

Definition

```
#include <atrac.h>
typedef struct {
        SceUInt32 size;
        SceUInt32 wordLength;
        SceUInt32 totalCh;
} SceAtracDecoderGroup;
```

Members

size	Size of the structure
wordLength	Number of PCM quantization bits
totalCh	Total number of channels that can be decoded at the same time

Description

This is the structure for libatrac decoder group.

This structure manages decoder groups for each audio codec type and can be used for the following purposes.

- Obtains the required memory size for creating a decoder group using `sceAtracQueryDecoderGroupMemSize()`
- Creates a decoder group using `sceAtracCreateDecoderGroup()`
- Obtains the created decoder group and usable decoder group information using `sceAtracGetDecoderGroupInfo()`

See Also

`sceAtracQueryDecoderGroupMemSize()`, `sceAtracCreateDecoderGroup()`, `sceAtracGetDecoderGroupInfo()`

SceAtracContentInfo

Content information structure

Definition

```
#include <atrac.h>
typedef struct {
        SceUInt32 size;
        SceUInt32 atracType;
        SceUInt32 channel;
        SceUInt32 samplingRate;
        SceInt32 endSample;
        SceInt32 loopStartSample;
        SceInt32 loopEndSample;
        SceUInt32 bitRate;
        SceUInt32 fixedEncBlockSize;
        SceUInt32 fixedEncBlockSample;
        SceUInt32 frameSample;
        SceUInt32 loopBlockOffset;
        SceUInt32 loopBlockSize;
} SceAtracContentInfo;
```

Members

size	Size of the structure
atracType	ATRAC™ type
channel	Number of channels
samplingRate	Sampling frequency
endSample	End sample
loopStartSample	Loop start sample
loopEndSample	Loop end sample
bitRate	Bit rate (in kbps)
fixedEncBlockSize	Fixed encode block size
fixedEncBlockSample	Number of fixed encode block samples
frameSample	Number of frame samples
loopBlockOffset	Start position of loop data
loopBlockSize	Size of loop data

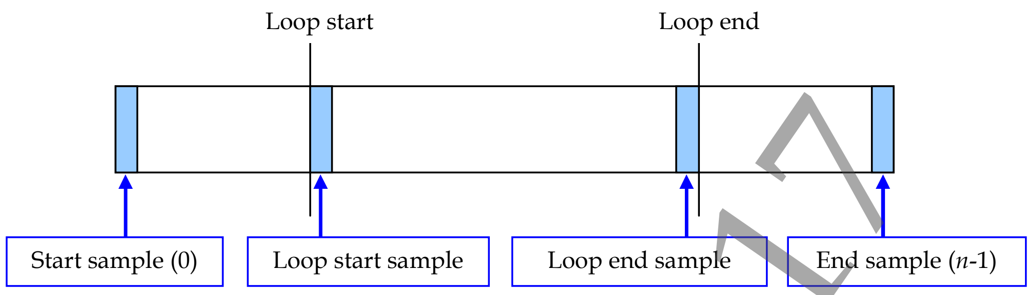
Description

This is the structure for libatrac content information.

The start sample, end sample, loop start sample, and loop end sample for the total number of *n* samples are shown in Figure 1.

In addition, -1 is set to the loop start sample and loop end sample for data without loop information.

Figure 1 Example of End and Loop Sample



See Also

`sceAtracGetContentInfo()`

SceAtracStreamInfo

Streaming information structure

Definition

```
#include <atrac.h>
typedef struct {
        SceUInt32 size;
        SceUChar8 *pWritePosition;
        SceUInt32 readPosition;
        SceUInt32 writableSize;
} SceAtracStreamInfo;
```

Members

size	Size of the structure
pWritePosition	Start address of writing to the buffer
readPosition	Reading position of audio data
writableSize	Maximum number of bytes writable to the buffer

Description

This structure is used to obtain the information on reading data of the buffer when performing streaming playback.

See Also

`sceAtracGetStreamInfo()`

Controlling Decoder Group

sceAtracQueryDecoderGroupMemSize

Inquire memory size required for creating a decoder group

Definition

```
#include <atrac.h>
SceInt32 sceAtracQueryDecoderGroupMemSize (
        SceUInt32 atracType,
        const SceAtracDecoderGroup *pDecoderGroup
)
```

Arguments

atracType	ATRAC™ type
pDecoderGroup	Pointer to decoder group structure

Return Values

Returns required memory size upon normal termination.

Returns one of the following error codes (negative value) upon error.

Value	(Number)	Description
SCE_ATRAC_ERROR_INVALID_POINTER	0x80630000	Invalid pointer argument
SCE_ATRAC_ERROR_INVALID_SIZE	0x80630001	Invalid size
SCE_ATRAC_ERROR_INVALID_WORD_LENGTH	0x80630002	Invalid PCM quantization size
SCE_ATRAC_ERROR_INVALID_TYPE	0x80630003	Invalid ATRAC™ type
SCE_ATRAC_ERROR_INVALID_TOTAL_CH	0x80630004	Invalid total number of channels

Description

This function inquires the memory size required for creating a decoder group.

For *atracType*, specify an ATRAC™ type to be used.

To *pDecoderGroup*, specify the pointer to the decoder group structure that is used to set the total number of channels and the number of PCM quantization bits to be used.
By calling this function, it is possible to obtain the memory size required for creating a decoder group.

Notes

This function is multi-thread safe.

Examples

```
SceAtracDecoderGroup decoderGroup;
int32_t memorySize;

decoderGroup.size = sizeof(SceAtracDecoderGroup);
decoderGroup.totalCh = SCE_ATRAC_AT9_MAX_TOTAL_CH;
decoderGroup.wordLength = SCE_ATRAC_WORD_LENGTH_16BITS;

// Inquire memory size required for creating a decoder group
memorySize = sceAtracQueryDecoderGroupMemSize(SCE_ATRAC_TYPE_AT9,
&decoderGroup);
if (memorySize < 0) {
        // Error handling
}
```

See Also

`SceAtracDecoderGroup`, `sceAtracCreateDecoderGroup()`, ATRAC™ Type, Number of PCM Quantization Bits

sceAtracCreateDecoderGroup

Create decoder group

Definition

```
#include <atrac.h>
SceInt32 sceAtracCreateDecoderGroup (
        SceUInt32 atracType,
        const SceAtracDecoderGroup *pDecoderGroup,
        void *pvWorkMem,
        SceInt32 initAudiodecFlag
)
```

Arguments

atracType	ATRAC™ type
pDecoderGroup	Pointer to decoder group structure
pvWorkMem	Start address of work memory (256-byte alignment)
initAudiodecFlag	Audio decoder initialization flag

Return Values

Returns `SCE_OK(0)` upon normal termination.

Returns one of the following error codes (negative value) upon error.

Value	(Number)	Description
`SCE_ATRAC_ERROR_INVALID_POINTER`	0x80630000	Invalid pointer argument
`SCE_ATRAC_ERROR_INVALID_SIZE`	0x80630001	Invalid size
`SCE_ATRAC_ERROR_INVALID_WORD_LENGTH`	0x80630002	Invalid PCM quantization size
`SCE_ATRAC_ERROR_INVALID_TYPE`	0x80630003	Invalid ATRAC™ type
`SCE_ATRAC_ERROR_INVALID_TOTAL_CH`	0x80630004	Invalid total number of channels
`SCE_ATRAC_ERROR_INVALID_ALIGNMENT`	0x80630005	Invalid alignment
`SCE_ATRAC_ERROR_ALREADY_CREATED`	0x80630006	Decoder group already created

Description

This function creates a decoder group.
Be sure to create a decoder group when using libatrac to perform decoding.

For *atracType*, specify an ATRAC™ type to be used.

To *pDecoderGroup*, specify the pointer to the decoder group structure that is used to set the total number of channels and the number of PCM quantization bits to be used.

To *pvWorkMem*, specify the start address of the work memory of the memory size obtained with `sceAtracQueryDecoderGroupMemSize()`. *pvWorkMem* must have a 256-byte alignment.

initAudiodecFlag is a flag to determine whether to call `sceAudiodecInitLibrary()` within libatrac. When `SCE_FALSE` is specified to *initAudiodecFlag*, the user must call `sceAudiodecInitLibrary()` before calling this function.

Notes

This function is not multi-thread safe. If it is called at the same time from different threads, the library may later malfunction even if this function terminates normally. Therefore, avoid simultaneous calls when programming.

Examples

```
SceAtracDecoderGroup decoderGroup;
int32_t memorySize;
int32_t returnCode;
uint8_t *pWorkMem;

decoderGroup.size = sizeof(SceAtracDecoderGroup);
decoderGroup.totalCh = SCE_ATRAC_AT9_MAX_TOTAL_CH;
decoderGroup.wordLength = SCE_ATRAC_WORD_LENGTH_16BITS;

// Inquire memory size required for creating a decoder group
memorySize = sceAtracQueryDecoderGroupMemSize(SCE_ATRAC_TYPE_AT9,
                                              &decoderGroup);
if (memorySize < 0) {
        // Error handling
}

// Allocate a 256-byte aligned work memory (pWorkMem)

// Create a decoder group
returnCode = sceAtracCreateDecoderGroup(SCE_ATRAC_TYPE_AT9,
                                        &decoderGroup,
                                        pWorkMem,
                                        SCE_TRUE);
if (returnCode < 0) {
        //Error handling
}
```

See Also

`SceAtracDecoderGroup`, `sceAtracQueryDecoderGroupMemSize()`, ATRAC™ Type, Number of PCM Quantization Bits

sceAtracDeleteDecoderGroup

Delete decoder group

Definition

```
#include <atrac.h>
SceInt32 sceAtracDeleteDecoderGroup (
        SceUInt32 atracType,
        SceInt32 termAudiodecFlag
)
```

Arguments

atracType	ATRAC™ type
termAudiodecFlag	Audio decoder end flag

Return Values

Returns `SCE_OK(0)` upon normal termination.

Returns one of the following error codes (negative value) upon error.

Value	(Number)	Description
`SCE_ATRAC_ERROR_INVALID_TYPE`	0x80630003	Invalid ATRAC™ type
`SCE_ATRAC_ERROR_NOT_CREATED`	0x80630007	Decoder group not created
`SCE_ATRAC_ERROR_REMAIN_VALID_HANDLE`	0x80630018	Valid handle remaining

Description

This function deletes a decoder group.

For *atracType*, specify an ATRAC™ type to be used.

termAudiodecFlag is a flag to determine whether to call `sceAudiodecTermLibrary()` within libatrac. When `SCE_FALSE` is specified to *termAudiodecFlag*, the user must call `sceAudiodecTermLibrary()`.

Note that `sceAtracReleaseHandle()` must be used to free all handles obtained with `sceAtracSetDataAndAcquireHandle()` when calling this function.

Notes

This function is not multi-thread safe. If it is called at the same time from different threads, the library may later malfunction even if this function terminates normally. Therefore, avoid simultaneous calls when programming.

Examples

```
int32_t returnCode;

returnCode = sceAtracDeleteDecoderGroup(SCE_ATRAC_TYPE_AT9, SCE_TRUE);
if (returnCode < 0) {
        // Error handling
}
```

See Also

`sceAtracCreateDecoderGroup()`, ATRAC™ Type

sceAtracGetDecoderGroupInfo

Get decoder group information

Definition

```
#include <atrac.h>
SceInt32 sceAtracGetDecoderGroupInfo (
        SceUInt32 atracType,
        SceAtracDecoderGroup *pCreatedDecoderGroup,
        SceAtracDecoderGroup *pAvailableDecoderGroup
)
```

Arguments

atracType	ATRAC™ type
pCreatedDecoderGroup	Pointer to structure storing information of created decoder group
pAvailableDecoderGroup	Pointer to structure storing information of usable decoder group

Return Values

Returns `SCE_OK(0)` upon normal termination.

Returns one of the following error codes (negative value) upon error.

Value	(Number)	Description
`SCE_ATRAC_ERROR_INVALID_POINTER`	0x80630000	Invalid pointer argument
`SCE_ATRAC_ERROR_INVALID_SIZE`	0x80630001	Invalid size
`SCE_ATRAC_ERROR_INVALID_TYPE`	0x80630003	Invalid ATRAC™ type
`SCE_ATRAC_ERROR_NOT_CREATED`	0x80630007	Decoder group not created

Description

This function obtains decoder group information.

This function can be used to obtain the decoder group information created with `sceAtracCreateDecoderGroup()` and the decoder group information that can be currently used.

Notes

This function is not multi-thread safe. If it is called at the same time from different threads, the library may later malfunction even if this function terminates normally. Therefore, avoid simultaneous calls when programming.

Examples

```
SceAtracDecoderGroup createdDecoderGroup, availableDecoderGroup;
int32_t returnCode;

createdDecoderGroup.size = sizeof(SceAtracDecoderGroup);
availableDecoderGroup.size = sizeof(SceAtracDecoderGroup);

// Obtain decoder group information
returnCode = sceAtracGetDecoderGroupInfo(SCE_ATRAC_TYPE_AT9,
                               &createdDecoderGroup,
&availableDecoderGroup);
if (returnCode < 0) {
        // Error handling
}
```

See Also

`sceAtracCreateDecoderGroup()`, ATRAC™ Type

Controlling ATRAC™ Handle

sceAtracSetDataAndAcquireHandle

Set the ATRAC9™ data to be input and ATRAC™ handle

Definition

```
#include <atrac.h>
SceInt32 sceAtracSetDataAndAcquireHandle (
        SceUChar8 *pMainBuffer,
        SceUInt32 readSize,
        SceUInt32 mainBufferSize
)
```

Arguments

pMainBuffer	Start address of main buffer (256-byte alignment)
readSize	Buffer read size
mainBufferSize	Main buffer size (multiple of 256)

Return Values

Returns ATRAC™ handle (>=0) upon normal termination.

Returns one of the following error codes (negative value) upon error.

Value	(Number)	Description
SCE_ATRAC_ERROR_INVALID_POINTER	0x80630000	Invalid pointer argument
SCE_ATRAC_ERROR_INVALID_ALIGNMENT	0x80630005	Invalid alignment
SCE_ATRAC_ERROR_NOT_CREATED	0x80630007	Decoder group not created
SCE_ATRAC_ERROR_SHORTAGE_OF_CH	0x80630008	Insufficient usable channels
SCE_ATRAC_ERROR_UNSUPPORTED_DATA	0x80630009	Unsupported data
SCE_ATRAC_ERROR_INVALID_DATA	0x8063000A	Invalid data
SCE_ATRAC_ERROR_READ_SIZE_IS_TOO_SMALL	0x8063000B	Set size too small
SCE_ATRAC_ERROR_READ_SIZE_OVER_BUFFER	0x8063000D	Invalid read size and buffer size
SCE_ATRAC_ERROR_MAIN_BUFFER_SIZE_IS_TOO_SMALL	0x8063000E	Main buffer size too small

Description

This function sets the ATRAC9™ data to be input and obtains ATRAC™ handle.

Specify a 256-byte aligned memory to `pMainBuffer`.

Specify the number of bytes to `mainBufferSize` in multiples of 256.

Note that `mainBufferSize` must be equal to or more than a buffer size that is the sum of the ATRAC9™ header size plus a size worth 48 frames. When the buffer size is small, sound skips occur. It is recommended to allocate a buffer of at least 50 KB.

For `readSize`, set a size that is equal to or more than the sum of the ATRAC9™ header size plus the size of `nBlockAlign`.

Notes

This function is not multi-thread safe. If it is called at the same time from different threads, the library may later malfunction even if this function terminates normally. Therefore, avoid simultaneous calls when programming.

Examples

```
#define MAIN_BUFFER_SIZE (50 * 1024)
int32_t atracHandle;
uint8_t *pMainBuffer;

// Create a decoder group

// Allocate a 256-byte aligned main buffer(pMainBuffer)

// Reading of data to pMainBuffer

// set the input data and obtain handle
atracHandle = sceAtracSetDataAndAcquireHandle(pMainBuffer,
                                              MAIN_BUFFER_SIZE,
                                              MAIN_BUFFER_SIZE);
if (atracHandle < 0) {
        // Error handling
}
```

See Also

`sceAtracCreateDecoderGroup()`, `sceAtracReleaseHandle()`, Alignment Size

sceAtracReleaseHandle

Free ATRAC™ handle

Definition

```
#include <atrac.h>
SceInt32 sceAtracReleaseHandle (
        SceInt32 atracHandle
)
```

Arguments

atracHandle	ATRAC™ handle

Return Values

Returns `SCE_OK(0)` upon normal termination.

Returns the following error code (negative value) upon error.

Value	(Number)	Description
SCE_ATRAC_ERROR_INVALID_HANDLE	0x8063000C	Invalid handle

Description

This function frees ATRAC™ handle.

The handle obtained with `sceAtracSetDataAndAcquireHandle()` must be freed using `sceAtracReleaseHandle()`.

Notes

This function is not multithread safe for the same handle. If this function is called at the same time from multiple threads with the same handle specified, the library may later malfunction even if this function terminates normally.
This function is multithread safe for differing handles.

Examples

```
int32_t atracHandle;
int32_t returnCode;

// Obtain atracHandle

// Free a handle
returnCode = sceAtracReleaseHandle(atracHandle);
if (returnCode < 0) {
        // Error handling
}
```

See Also

`sceAtracDeleteDecoderGroup(),sceAtracSetDataAndAcquireHandle()`

Decoding

sceAtracDecode

Execute decoding process

Definition

```
#include <atrac.h>
SceInt32 sceAtracDecode (
        SceInt32 atracHandle,
        void *pOutputBuffer,
        SceUInt32 *pOutputSamples,
        SceUInt32 *pDecoderStatus
)
```

Arguments

atracHandle	ATRAC™ handle
pOutputBuffer	Pointer indicating decoded result output destination (2-byte alignment)
pOutputSamples	Pointer to variable for storing number of output samples per channel
pDecoderStatus	Pointer to variable for storing decoder state identifier

Return Values

Returns `SCE_OK(0)` upon normal termination.

Returns one of the following error codes (negative value) upon error.

Value	(Number)	Description
SCE_ATRAC_ERROR_INVALID_POINTER	0x80630000	Invalid pointer argument
SCE_ATRAC_ERROR_INVALID_ALIGNMENT	0x80630005	Invalid alignment
SCE_ATRAC_ERROR_INVALID_HANDLE	0x8063000C	Invalid handle
SCE_ATRAC_ERROR_DATA_SHORTAGE_IN_BUFFER	0x80630010	Insufficient data in buffer
SCE_ATRAC_ERROR_ALL_DATA_WAS_DECODED	0x80630011	All data decoded
SCE_ATRAC_ERROR_NEED_SUB_BUFFER	0x80630014	Sub buffer not set

Description

This function decodes and outputs an audio data.

The audio data is output to the output buffer indicated with *pOutputBuffer* in interleave format (whereby channel data is aligned with each sample).

The output buffer must have a 256-byte alignment and have a size in multiples of 256 bytes. However, when providing a large output buffer and continually writing results that have been decoded multiple times, the address specified to *pOutputBuffer* may conform to a 2-byte alignment and does not have to have a 256-byte alignment.

pOutputSamples stores the number of output samples per channel.

pDecoderStatus stores the decoder state.

The number of output samples can be set with `sceAtracSetOutputSamples()`.

If the number of remaining samples is less than the number of output samples, only the number of remaining samples is output.

Notes

This function is not multithread safe for the same handle. If this function is called at the same time from multiple threads with the same handle specified, the library may later malfunction even if this function terminates normally.

This function is multithread safe for differing handles.

Examples

```
int32_t returnCode;
int32_t atracHandle;
SceAtracStreamInfo streamInfo;
int16_t *pOutputBuffer;
uint32_t outputSamples, decoderStatus;

// Obtain atracHandle

// Allocate output buffer (pOutputBuffer)

returnCode = sceAtracDecode(atracHandle,
                            pOutputBuffer,
                            &outputSamples,
                             &decoderStatus);
if (returnCode < 0) {
        // Error handling
}
```

See Also

`sceAtracSetDataAndAcquireHandle()`, `sceAtracGetContentInfo()`, `sceAtracGetOutputSamples()`, Alignment Size

Streaming Process

sceAtracGetStreamInfo

Get streaming information

Definition

```
#include <atrac.h>
SceInt32 sceAtracGetStreamInfo (
        SceInt32 atracHandle,
        SceAtracStreamInfo *pStreamInfo
)
```

Arguments

atracHandle	ATRAC™ handle
pStreamInfo	Pointer to streaming information structure

Return Values

Returns `SCE_OK(0)` upon normal termination.

Returns one of the following error codes (negative value) upon error.

Value	(Number)	Description
SCE_ATRAC_ERROR_INVALID_POINTER	0x80630000	Invalid pointer argument
SCE_ATRAC_ERROR_INVALID_SIZE	0x80630001	Invalid size
SCE_ATRAC_ERROR_INVALID_HANDLE	0x8063000C	Invalid handle

Description

This function obtains streaming information.

For an application, after calling this function, use the streaming information to read the ATRAC9™ data. Next, use `sceAtracAddStreamData()` to send notification of the addition of ATRAC9™ data to the library.

Notes

This function is not multithread safe for the same handle. If this function is called at the same time from multiple threads with the same handle specified, the library may later malfunction even if this function terminates normally.

This function is multithread safe for differing handles.

Examples

```
int32_t returnCode;
int32_t atracHandle;
SceAtracStreamInfo streamInfo;

// Obtain atracHandle

// Obtain streaming information
streamInfo.size = sizeof(SceAtracStreamInfo);

returnCode = sceAtracGetStreamInfo(atracHandle, &streamInfo);
if (returnCode < 0) {
        // Error handling
}
```

See Also

`SceAtracStreamInfo`, `sceAtracSetDataAndAcquireHandle()`, `sceAtracAddStreamData()`

sceAtracAddStreamData

Send notification of stream data addition

Definition

```
#include <atrac.h>
SceInt32 sceAtracAddStreamData (
        SceInt32 atracHandle,
        SceUInt32 addSize
)
```

Arguments

atracHandle	ATRAC™ handle
addSize	Size of ATRAC9™ data added to buffer (bytes)

Return Values

Returns `SCE_OK(0)` upon normal termination.

Returns one of the following error codes (negative value) upon error.

Value	(Number)	Description
`SCE_ATRAC_ERROR_INVALID_HANDLE`	0x8063000C	Invalid handle
`SCE_ATRAC_ERROR_ADDED_DATA_IS_TOO_BIG`	0x80630013	Invalid size of added data

Description

This function sends notification to the library that ATRAC9™ data has been added to the input buffer during streaming playback.

Read the ATRAC9™ data based on the streaming information obtained with `sceAtracGetStreamInfo()` and call this function.

Notes

This function is not multithread safe for the same handle. If this function is called at the same time from multiple threads with the same handle specified, the library may later malfunction even if this function terminates normally.

This function is multithread safe for differing handles.

Examples

```
int32_t returnCode;
int32_t atracHandle;
SceAtracStreamInfo streamInfo;

// Obtain atracHandle

// Obtain streaming information
streamInfo.size = sizeof(SceAtracStreamInfo);

returnCode = sceAtracGetStreamInfo(atracHandle, &streamInfo);
if (returnCode < 0) {
        // Error handling
}

// Seek file read position

// Read data

// Send notification of stream data addition
returnCode = sceAtracAddStreamData(atracHandle, streamInfo.writableSize);
if (returnCode < 0) {
        // Error handling
}
```

See Also

`SceAtracStreamInfo`, `sceAtracSetDataAndAcquireHandle()`, `sceAtracGetStreamInfo()`

Sub Buffer Process

sceAtracIsSubBufferNeeded

Check necessity of sub buffer

Definition

```
#include <atrac.h>
SceInt32 sceAtracIsSubBufferNeeded (
        SceInt32 atracHandle
)
```

Arguments

atracHandle	ATRAC™ handle

Return Values

Returns a positive number if a sub buffer is required and 0 if not required.

Returns the following error code (negative value) upon error.

Value	(Number)	Description
`SCE_ATRAC_ERROR_INVALID_HANDLE`	0x8063000C	Invalid handle

Description

This function checks whether setting of a sub buffer is required.

A sub buffer may be required when the content being played back is streamed using data with an epilogue (data with audio after a loop).

A sub buffer is used to ensure smooth playback when retaining data of an epilogue section and transitioning to an epilogue section after stopping loop playback.

Use `sceAtracGetSubBufferInfo()` to obtain the information related to the data stored in the sub buffer.

Notes

This function is not multithread safe for the same handle. If this function is called at the same time from multiple threads with the same handle specified, the library may later malfunction even if this function terminates normally.

This function is multithread safe for differing handles.

Examples

```
int32_t returnCode;
int32_t atracHandle;

// Obtain atracHandle

if (sceAtracIsSubBufferNeeded(atracHandle)) {

        // Sub buffer set processing
}
```

See Also

`sceAtracSetDataAndAcquireHandle(),sceAtracGetSubBufferInfo()`

sceAtracGetSubBufferInfo

Get sub buffer information

Definition

```
#include <atrac.h>
SceInt32 sceAtracGetSubBufferInfo (
        SceInt32 atracHandle,
        SceUInt32 *pReadPosition,
        SceUInt32 *pMinSubBufferSize,
        SceUInt32 *pDataSize
)
```

Arguments

atracHandle	ATRAC™ handle
pReadPosition	Pointer to variable for storing read start position on file
pMinSubBufferSize	Pointer to variable for storing minimum sub buffer size (multiple of 256)
pDataSize	Data size readable to sub buffer

Return Values

Returns `SCE_OK(0)` upon normal termination.

Returns one of the following error codes (negative value) upon error.

Value	(Number)	Description
`SCE_ATRAC_ERROR_INVALID_POINTER`	0x80630000	Invalid pointer argument
`SCE_ATRAC_ERROR_INVALID_HANDLE`	0x8063000C	Invalid handle
`SCE_ATRAC_ERROR_NO_NEED_SUB_BUFFER`	0x80630016	Sub buffer not required

Description

This function obtains the ATRAC9™ data information to be read to the sub buffer.

A sub buffer may be required when the content being played back is streamed using data with an epilogue (data with audio after a loop).

When a sub buffer is required, call this function, read the data of an epilogue section from the file, according to the obtained information, and write the data to the memory area allocated by the application side. Next, call `sceAtracSetSubBuffer()` to register that area as a sub buffer.

Note that the minimum sub buffer size is in multiples of 256, therefore, the minimum sub buffer size may be larger than the readable data size.

Notes

This function is not multithread safe for the same handle. If this function is called at the same time from multiple threads with the same handle specified, the library may later malfunction even if this function terminates normally.

This function is multithread safe for differing handles.

Examples

```
int32_t returnCode;
int32_t atracHandle;

// Obtain atracHandle

if (sceAtracIsSubBufferNeeded(atracHandle)) {
        uint32_t readPosition, minSubBufferSize, dataSize;

        // Obtain sub buffer information
        returnCode = sceAtracGetSubBufferInfo(atracHandle,
                                              &readPosition,
                                              &minSubBufferSize,
                                              &dataSize);
        if (returnCode < 0) {
              // Error handling
        }
        // Sub buffer set processing
}
```

See Also

`sceAtracSetDataAndAcquireHandle(),sceAtracSetSubBuffer()`

sceAtracSetSubBuffer

Set sub buffer

Definition

```
#include <atrac.h>
SceInt32 sceAtracSetSubBuffer (
        SceInt32 atracHandle,
        SceUChar8 *pSubBuffer,
        SceUInt32 subBufferSize
)
```

Arguments

atracHandle	ATRAC™ handle
pSubBuffer	Pointer to start of sub buffer (256-byte alignment)
subBufferSize	Sub buffer size (multiple of 256)

Return Values

Returns `SCE_OK(0)` upon normal termination.

Returns one of the following error codes (negative value) upon error.

Value	(Number)	Description
SCE_ATRAC_ERROR_INVALID_ALIGNMENT	0x80630005	Invalid alignment
SCE_ATRAC_ERROR_INVALID_HANDLE	0x8063000C	Invalid handle
SCE_ATRAC_ERROR_SUB_BUFFER_SIZE_IS_TOO_SMALL	0x8063000F	Sub buffer size too small
SCE_ATRAC_ERROR_NO_NEED_SUB_BUFFER	0x80630016	Sub buffer not required

Description

This function registers the sub buffer to the library.

A sub buffer may be required when the content being played back is streamed using data with an epilogue (data with audio after a loop).

A large buffer size is not required as the sub buffer is only used temporarily so that data is not interrupted when transitioning to the epilogue section. As a function specification, the buffer size must be at least worth 16 frames. However, if the reading of ATRAC9™ data does not catch up, the sound may be interrupted, so the buffer must have a sufficient size.

Notes

This function is not multithread safe for the same handle. If this function is called at the same time from multiple threads with the same handle specified, the library may later malfunction even if this function terminates normally.

This function is multithread safe for differing handles.

Examples

```
int32_t returnCode;
int32_t atracHandle;

// Obtain atracHandle

if (sceAtracIsSubBufferNeeded(atracHandle)) {
        uint32_t readPosition, minSubBufferSize, dataSize;
        uint8_t *pSubBuffer;

        // Obtain sub buffer information
        returnCode = sceAtracGetSubBufferInfo(atracHandle,
                                               &readPosition,
                                               &minSubBufferSize,
                                               &dataSize);
        if (returnCode < 0) {
              // Error handling
        }

        // Allocate a 256-byte aligned sub buffer (pSubBuffer)

        // Seek file read position

        // Read data to sub buffer data

        // Set sub buffer
        returnCode = sceAtracSetSubBuffer(atracHandle,
                                           pSubBuffer,
                                           minSubBufferSize);

}
```

See Also

`sceAtracSetDataAndAcquireHandle()`, `sceAtracGetSubBufferInfo()`, Alignment Size

Decoder Information Setting Process

sceAtracSetLoopNum

Reset number of loops

Definition

```
#include <atrac.h>
SceInt32 sceAtracSetLoopNum (
        SceInt32 atracHandle,
        SceInt32 loopNum
)
```

Arguments

atracHandle	ATRAC™ handle
loopNum	Number of loops

Return Values

Returns `SCE_OK(0)` upon normal termination.

Returns one of the following error codes (negative value) upon error.

Value	(Number)	Description
SCE_ATRAC_ERROR_INVALID_HANDLE	0x8063000C	Invalid handle
SCE_ATRAC_ERROR_INVALID_LOOP_STATUS	0x80630017	Invalid loop state
SCE_ATRAC_ERROR_INVALID_LOOP_NUM	0x80630030	Invalid number of loops

Description

This function resets the number of loops.

The following values can be set to *loopNum*.

Value	Description
0	Does not perform loop playback
> 0	Performs loop playback for the specified number of times
SCE_ATRAC_INFINITE_LOOP_NUM(-1)	Performs infinite loop playback

The following two conditions must be satisfied to reset the number of loops.

- Loop information is set to the input data. (For the setting method, refer to the "at9tool User's Guide" document.)
- The loop state identifier obtained with `sceAtracGetLoopInfo()` is `SCE_ATRAC_LOOP_STATUS_RESETABLE_PART`

Notes

This function is not multithread safe for the same handle. If this function is called at the same time from multiple threads with the same handle specified, the library may later malfunction even if this function terminates normally.

This function is multithread safe for differing handles.

Examples

```
int32_t returnCode;
int32_t atracHandle;

// Obtain atracHandle

// Set number of loops
returnCode = sceAtracSetLoopNum(atracHandle, 3);
if (returnCode < 0) {
        // Error handling
}
```

See Also

`sceAtracSetDataAndAcquireHandle()`, `sceAtracGetLoopInfo()`, Loop State Identifier

sceAtracSetOutputSamples

Set number of output samples

Definition

```
#include <atrac.h>
SceInt32 sceAtracSetOutputSamples (
        SceInt32 atracHandle,
        SceUInt32 outputSamples
)
```

Arguments

atracHandle	ATRAC™ handle
outputSamples	Number of output samples

Return Values

Returns `SCE_OK(0)` upon normal termination.

Returns one of the following error codes (negative value) upon error.

Value	(Number)	Description
`SCE_ATRAC_ERROR_INVALID_HANDLE`	0x8063000C	Invalid handle
`SCE_ATRAC_ERROR_INVALID_MAX_OUTPUT_SAMPLES`	0x80630012	Invalid number of output samples

Description

This function sets the number of the output samples.

Use of this function may set the number of output samples per channel when `sceAtracDecode()` is called.

The number of output samples has the following limitations.

- Must be greater than 0
- Multiple of the number of frame samples
- Does not exceed the maximum number of output samples (`SCE_ATRAC_MAX_OUTPUT_SAMPLES`)
- Does not exceed maximum number of output frames x number of frame samples

If the number of remaining samples is equal to or less than the number of output samples, the number of output samples of `sceAtracDecode()` is the number of remaining samples.

In addition, the number of output samples immediately after creating a handle with `sceAtracSetDataAndAcquireHandle()` is the number of frame samples.

Notes

This function is not multithread safe for the same handle. If this function is called at the same time from multiple threads with the same handle specified, the library may later malfunction even if this function terminates normally.

This function is multithread safe for differing handles.

Examples

```
int32_t returnCode;
int32_t atracHandle;

// Obtain atracHandle

// Set the number of the output samples
returnCode = sceAtracSetOutputSamples(atracHandle,
                                      SCE_ATRAC_MAX_OUTPUT_SAMPLES);
if (returnCode < 0) {
        // Error handling
}
```

See Also

`sceAtracSetDataAndAcquireHandle()`, `sceAtracGetOutputSamples()`, Maximum Number of Output Samples, Maximum Number of Output Frames

sceAtracResetNextOutputPosition

Reset playback position

Definition

```
#include <atrac.h>
SceInt32 sceAtracResetNextOutputPosition(
        SceInt32 atracHandle,
        SceUInt32 resetSample
)
```

Arguments

atracHandle	ATRAC™ handle
resetSample	Output sample position to be reset (sample)

Return Values

Returns `SCE_OK(0)` upon normal termination.

Returns one of the following error codes (negative value) upon error.

Value	(Number)	Description
`SCE_ATRAC_ERROR_INVALID_HANDLE`	0x8063000C	Invalid handle
`SCE_ATRAC_ERROR_NEED_SUB_BUFFER`	0x80630014	Sub buffer not set
`SCE_ATRAC_ERROR_INVALID_SAMPLE`	0x80630015	Invalid sample

Description

This function changes the playback position.

To *resetSample*, specify the new playback position using the sample position with the data start sample at 0. In other words, when the total number of samples is the data of *n* samples, the sample position has a range of 0 to *n*-1.

In addition, when the playback position is changed during streaming playback, the data in the buffer is emptied.

Therefore, when the playback position is changed during streaming playback, use `sceAtracGetStreamInfo()` and `sceAtracAddStreamData()` to add data.

Notes

This function is not multithread safe for the same handle. If this function is called at the same time from multiple threads with the same handle specified, the library may later malfunction even if this function terminates normally.

This function is multithread safe for differing handles.

Examples

```
SceInt32 atracHandle;
SceInt32 internalError;

// Obtain atracHandle

// Reset output sample position
returnCode = sceAtracResetNextOutputPosition(atracHandle,0);
if (returnCode < 0) {
        // Error handling
}
```

See Also

`sceAtracSetDataAndAcquireHandle(),sceAtracGetNextOutputPosition()`

Obtaining Decoder Information

sceAtracGetContentInfo

Get content information

Definition

```
#include <atrac.h>
SceInt32 sceAtracGetContentInfo (
        SceInt32 atracHandle,
        SceAtracContentInfo*pContentInfo
)
```

Arguments

atracHandle	ATRAC™ handle
pContentInfo	Pointer to content information structure

Return Values

Returns `SCE_OK(0)` upon normal termination.

Returns one of the following error codes (negative value) upon error.

Value	(Number)	Description
SCE_ATRAC_ERROR_INVALID_POINTER	0x80630000	Invalid pointer argument
SCE_ATRAC_ERROR_INVALID_SIZE	0x80630001	Invalid size
SCE_ATRAC_ERROR_INVALID_HANDLE	0x8063000C	Invalid handle

Description

This function obtains content information.

Notes

This function is not multithread safe for the same handle. If this function is called at the same time from multiple threads with the same handle specified, the library may later malfunction even if this function terminates normally.

This function is multithread safe for differing handles.

Examples

```
int32_t returnCode;
int32_t atracHandle;
SceAtracContentInfo contentInfo;

// Obtain atracHandle

// Obtain content information
contentInfo.size = sizeof(SceAtracContentInfo);
returnCode = sceAtracGetContentInfo(atracHandle, &contentInfo);
if (returnCode < 0) {
        // Error handling
}
```

See Also

`SceAtracContentInfo`, `sceAtracSetDataAndAcquireHandle()`

sceAtracGetLoopInfo

Get loop information

Definition

```
#include <atrac.h>
SceInt32 sceAtracGetLoopInfo(
        SceInt32 atracHandle,
        SceInt32 *pLoopNum,
        SceUInt32 *pLoopStatus
)
```

Arguments

atracHandle	ATRAC™ handle
pLoopNum	Pointer to variable where remaining number of loops is stored
pLoopStatus	Pointer to variable where loop state identifier is stored

Return Values

Returns `SCE_OK(0)` upon normal termination.

Returns one of the following error codes (negative value) upon error.

Value	(Number)	Description
SCE_ATRAC_ERROR_INVALID_POINTER	0x80630000	Invalid pointer argument
SCE_ATRAC_ERROR_INVALID_HANDLE	0x8063000C	Invalid handle

Description

This function obtains the remaining number of loops and the loop state (playback position) of the input data set currently .

The following values are set to **pLoopNum*.

Value	Description
0	Does not perform loop playback
> 0	Performs loop playback for the specified number of times
SCE_ATRAC_INFINITE_LOOP_NUM(-1)	Performs infinite loop playback

The loop state identifier is set to **pLoopStatus*.

Notes

This function is not multithread safe for the same handle. If this function is called at the same time from multiple threads with the same handle specified, the library may later malfunction even if this function terminates normally.

This function is multithread safe for differing handles.

Examples

```
int32_t returnCode;
int32_t atracHandle;
int32_t loopNum;
uint32_t loopStatus;

// Obtain atracHandle

// Obtain loop information
returnCode = sceAtracGetLoopInfo (atracHandle, &loopNum, &loopStatus);
if (returnCode < 0) {
        // Error handling
}
```

See Also

`sceAtracSetDataAndAcquireHandle()`, `sceAtracSetLoopNum()`, Loop State Identifier

sceAtracGetOutputSamples

Get number of output samples

Definition

```
#include <atrac.h>
SceInt32 sceAtracGetOutputSamples (
        SceInt32 atracHandle,
        SceUInt32 *pOutputSamples
)
```

Arguments

atracHandle	ATRAC™ handle
pOutputSamples	Pointer to variable for storing number of output samples

Return Values

Returns `SCE_OK(0)` upon normal termination.

Returns one of the following error codes (negative value) upon error.

Value	(Number)	Description
`SCE_ATRAC_ERROR_INVALID_POINTER`	0x80630000	Invalid pointer argument
`SCE_ATRAC_ERROR_INVALID_HANDLE`	0x8063000C	Invalid handle

Description

This function obtains the number of the output samples.

With this function, you can obtain the number of output samples per channel when `sceAtracDecode()` is called.
If the number of remaining samples is equal to or less than the number of output samples, the number of output samples of `sceAtracDecode()` is the number of remaining samples.
In addition, the number of output samples immediately after creating a handle with `sceAtracSetDataAndAcquireHandle()` is the number of frame samples.

Notes

This function is not multithread safe for the same handle. If this function is called at the same time from multiple threads with the same handle specified, the library may later malfunction even if this function terminates normally.
This function is multithread safe for differing handles.

Examples

```
int32_t returnCode;
int32_t atracHandle;
uint32_t outputSamples;

// Obtain atracHandle

// Obtain the number of the output samples
returnCode = sceAtracGetOutputSamples (atracHandle, &outputSamples);
if (returnCode < 0) {
        // Error handling
        }
```

See Also

`sceAtracSetDataAndAcquireHandle(),sceAtracSetOutputSamples()`

sceAtracGetNextOutputPosition

Get output sample position

Definition

```
#include <atrac.h>
SceInt32 sceAtracGetNextOutputPosition (
        SceInt32 atracHandle,
        SceUInt32 *pNextOutputSample
)
```

Arguments

atracHandle	ATRAC™ handle
pNextOutputSample	Pointer to variable for storing next output position

Return Values

Returns `SCE_OK(0)` upon normal termination.

Returns one of the following error codes (negative value) upon error.

Value	(Number)	Description
SCE_ATRAC_ERROR_INVALID_POINTER	0x80630000	Invalid pointer argument
SCE_ATRAC_ERROR_INVALID_HANDLE	0x8063000C	Invalid handle
SCE_ATRAC_ERROR_ALL_DATA_WAS_DECODED	0x80630011	All data decoded

Description

This function obtains the output sample position.

With this function, you can obtain the start sample position output when `sceAtracDecode()` is called the next time.

The sample position is counted with the data start sample as 0. In other words, when the total number of samples is the data of *n* samples, the sample position has a range of 0 to *n*-1.

Notes

This function is not multithread safe for the same handle. If this function is called at the same time from multiple threads with the same handle specified, the library may later malfunction even if this function terminates normally.

This function is multithread safe for differing handles.

Examples

```
int32_t returnCode;
int32_t atracHandle;
uint32_t nextOutputSample;

// Obtain atracHandle

// Obtain output position
returnCode = sceAtracGetNextOutputPosition (atracHandle, &nextOutputSample);
if (returnCode < 0) {
        // Error handling
}
```

See Also

`sceAtracSetDataAndAcquireHandle()`, `sceAtracResetNextOutputPosition()`

sceAtracGetRemainSamples

Get number of remaining samples

Definition

```
#include <atrac.h>
SceInt32 sceAtracGetRemainSamples(
        SceInt32 atracHandle,
        SceLong64 *pRemainSamples
)
```

Arguments

atracHandle	ATRAC™ handle
pRemainSamples	Pointer to variable for storing number of remaining samples

Return Values

Returns `SCE_OK(0)` upon normal termination.

Returns one of the following error codes (negative value) upon error.

Value	(Number)	Description
SCE_ATRAC_ERROR_INVALID_POINTER	0x80630000	Invalid pointer argument
SCE_ATRAC_ERROR_INVALID_HANDLE	0x8063000C	Invalid handle

Description

This function obtains the number of remaining samples.

By calling this function, you can obtain the number of remaining samples from the current playback position to the last output.
When playing back an infinite loop of data with loop information, the number of infinite samples is set to *pRemainSamples*.

Notes

This function is not multithread safe for the same handle. If this function is called at the same time from multiple threads with the same handle specified, the library may later malfunction even if this function terminates normally.
This function is multithread safe for differing handles.

Examples

```
int32_t returnCode;
int32_t atracHandle;
int64_t remainSamples;

// Obtain atracHandle

// Get number of remaining samples
returnCode = sceAtracGetRemainSamples (atracHandle, &remainSamples);
if (returnCode < 0) {
        // Error handling
}
```

See Also

`sceAtracSetDataAndAcquireHandle()`

sceAtracGetOutputableSamples

Get number of outputable samples

Definition

```
#include <atrac.h>
SceInt32 sceAtracGetOutputableSamples (
        SceInt32 atracHandle,
        SceLong64 *pOutputableSamples
)
```

Arguments

atracHandle	ATRAC™ handle
pOutputableSamples	Pointer to variable for storing number of outputable samples

Return Values

Returns `SCE_OK(0)` upon normal termination.

Returns one of the following error codes (negative value) upon error.

Value	(Number)	Description
`SCE_ATRAC_ERROR_INVALID_POINTER`	0x80630000	Invalid pointer argument
`SCE_ATRAC_ERROR_INVALID_HANDLE`	0x8063000C	Invalid handle

Description

This function obtains the number of outputable samples.

By calling this function, you can obtain the number of outputable samples from the data being read to the buffer.

When playing back an infinite loop of data with loop information, the number of infinite samples is set to *`pOutputableSamples`*.

Notes

This function is not multithread safe for the same handle. If this function is called at the same time from multiple threads with the same handle specified, the library may later malfunction even if this function terminates normally.

This function is multithread safe for differing handles.

Examples

```
int32_t returnCode;
int32_t atracHandle;
int64_t outputableSamples;

// Obtain atracHandle

// Get number of outputable samples
returnCode = sceAtracGetOutputableSamples(atracHandle, & outputableSamples);
if (returnCode < 0) {
        // Error handling
}
```

See Also

`sceAtracSetDataAndAcquireHandle()`

sceAtracGetDecoderStatus

Get decoder state

Definition

```
#include <atrac.h>
SceInt32 sceAtracGetDecoderStatus (
        SceInt32 atracHandle,
        SceUInt32 *pDecoderStatus
)
```

Arguments

atracHandle	ATRAC™ handle
pDecoderStatus	Pointer to variable for storing decoder state identifier

Return Values

Returns `SCE_OK(0)` upon normal termination.

Returns one of the following error codes (negative value) upon error.

Value	(Number)	Description
SCE_ATRAC_ERROR_INVALID_POINTER	0x80630000	Invalid pointer argument
SCE_ATRAC_ERROR_INVALID_HANDLE	0x8063000C	Invalid handle

Description

This function obtains the decoder state.

Notes

This function is not multithread safe for the same handle. If this function is called at the same time from multiple threads with the same handle specified, the library may later malfunction even if this function terminates normally.

This function is multithread safe for differing handles.

Examples

```
int32_t returnCode;
int32_t atracHandle;
uint32_t decoderStatus;

// Obtain atracHandle

// Obtain decoder state
returnCode = sceAtracGetDecoderStatus(atracHandle, &decoderStatus);
if (returnCode < 0) {
        // Error handling
}
```

See Also

`sceAtracSetDataAndAcquireHandle()`, Decoder State Identifier

sceAtracGetVacantSize

Get free buffer size

Definition

```
#include <atrac.h>
SceInt32 sceAtracGetVacantSize (
        SceInt32 atracHandle,
        SceUInt32 *pVacantSize
)
```

Arguments

atracHandle	ATRAC™ handle
pVacantSize	Pointer to variable for storing free buffer size

Return Values

Returns `SCE_OK(0)` upon normal termination.

Returns one of the following error codes (negative value) upon error.

Value	(Number)	Description
`SCE_ATRAC_ERROR_INVALID_POINTER`	0x80630000	Invalid pointer argument
`SCE_ATRAC_ERROR_INVALID_HANDLE`	0x8063000C	Invalid handle

Description

This function obtains the free buffer size.

By calling this function, you can obtain the free buffer size during streaming playback and determine how many bytes of data can be added.

Data may not be read to the end of the buffer to prevent unnecessary copying of the memory and a drop in performance during streaming playback of libatrac. As a result, when this function is used to check only the free buffer size, and then data is added from the previous buffer writing end position to the end of the buffer, libatrac may not operate properly.

To add data, read the data using the information of `sceAtracGetStreamInfo()`.

Notes

This function is not multithread safe for the same handle. If this function is called at the same time from multiple threads with the same handle specified, the library may later malfunction even if this function terminates normally.

This function is multithread safe for differing handles.

Examples

```
int32_t returnCode;
int32_t atracHandle;
uint32_t vacantSize;

// Obtain atracHandle

// Get free buffer size
returnCode = sceAtracGetVacantSize(atracHandle, &vacantSize);
if (returnCode < 0) {
        // Error handling
}
```

See Also

sceAtracSetDataAndAcquireHandle()

sceAtracGetInternalError

Get Codec Engine internal errors

Definition

```
#include <atrac.h>
SceInt32 sceAtracGetInternalError(
        SceInt32 atracHandle,
        SceInt32 *pInternalError
)
```

Arguments

atracHandle	ATRAC™ handle
pInternalError	Pointer to internal error variables

Return Values

Returns `SCE_OK(0)` as the value of the function upon normal termination.

Returns one of the following error codes (negative value) upon error.

Value	(Number)	Description
`SCE_ATRAC_ERROR_INVALID_POINTER`	0x80630000	Invalid pointer argument
`SCE_ATRAC_ERROR_INVALID_HANDLE`	0x8063000C	Invalid handle

Description

This function obtains Codec Engine internal errors.

By calling this function, details can be obtained regarding `SCE_AUDIODEC_ERROR_API_FAIL` internal errors within the Codec Engine.

This function is provided for supporting debugging. Programming that uses data obtained with this function to modify controls is not recommended.

`SCE_AUDIODEC_ERROR_API_FAIL` may occur in the libatrac library due to the following two reasons:

- There is an error in the encoded input data.
  Use the at9tool to decode the data, and check for any errors.
- After the data was read, it became corrupted in the application.
  Check whether the data read to the input buffer was overwritten.

Notes

This function is not multithread safe for the same handle. If this function is called at the same time from multiple threads with the same handle specified, the library may later malfunction even if this function terminates normally.

This function is multithread safe for differing handles.

Examples

```
SceInt32 atracHandle;
SceInt32 internalError;

// Obtain atracHandle

// Obtain error within the streaming library
returnCode = sceAtracGetInternalError(atracHandle, &internalError);
if (returnCode < 0) {
        // Error handling
}
```

See Also

```
sceAtracSetDataAndAcquireHandle()
```

Constants

Alignment Size

Alignment size

Definition

Value	(Number)	Description
`SCE_ATRAC_ALIGNMENT_SIZE`	0x100U	Alignment size

Description

This is the alignment size required for data accessed by the ATRAC9™ decoder.

Allocate memory with the correct alignment size for the following buffers:

- Work buffer given to `sceAtracCreateDecoderGroup()`
- Main buffer given to `sceAtracSetDataAndAcquireHandle()`
- Sub buffer given to `sceAtracSetSubBuffer()`
- Output buffer given to `sceAtracDecode()`

ATRAC™ Type

ATRAC™ type

Definition

Value	(Number)	Description
`SCE_ATRAC_TYPE_AT9`	0x2003U	ATRAC9™

Description

This is an identifier that indicates the type of ATRAC™.

When calling `sceAtracQueryDecoderGroupMemSize()`, `sceAtracCreateDecoderGroup()`, `sceAtracDeleteDecoderGroup()`, or `sceAtracGetDecoderGroupInfo()`, specify this identifier.

Maximum Value for the Total Number of Channels

Maximum value for the total number of channels

Definition

Value	(Number)	Description
`SCE_ATRAC_AT9_MAX_TOTAL_CH`	16	Maximum value for the total number of ATRAC9™ channels that can be decoded by libatrac at the same time

Description

This identifier indicates the maximum value for the total number of ATRAC9™ channels.

When specifying the *`totalCh`* variable in the `SceAtracDecoderGroup` structure, ensure that it does not exceed this value.

Number of PCM Quantization Bits

Number of PCM quantization bits

Definition

Value	(Number)	Description
`SCE_ATRAC_WORD_LENGTH_16BITS`	16	16 bits

Description

This identifier indicates the number of PCM quantization bits for libatrac.
Set this identifier to the *`wordLength`* variable in the `SceAtracDecoderGroup` structure.

Maximum Number of Channels

Maximum number of channels

Definition

Value	(Number)	Description
`SCE_ATRAC_AT9_MAX_CH_IN_DECODER`	2	Maximum number of channels for ATRAC9™ decoders

Description

This identifier indicates the maximum number of channels per stream for ATRAC9™.

Maximum Number of Frame Samples

Maximum number of frame samples

Definition

Value	(Number)	Description
`SCE_ATRAC_AT9_MAX_FRAME_SAMPLES`	256	Maximum number of frame samples for ATRAC9™

Description

This identifier indicates the maximum number of frame samples for each ATRAC9™ channel.

Maximum Number of Output Samples

Maximum number of output samples

Definition

Value	(Number)	Description
`SCE_ATRAC_MAX_OUTPUT_SAMPLES`	2048	Maximum number of output samples

Description

This identifier indicates the maximum number of output samples for each channel output by `sceAtracDecode()`.

Maximum Number of Output Frames

Maximum number of output frames

Definition

Value	(Number)	Description
`SCE_ATRAC_AT9_MAX_OUTPUT_FRAMES`	8	Maximum number of output frames

Description

This identifier indicates the maximum number of output frames for each ATRAC9™ channel.

You can specify a value up to the identifier x the number of frame samples to *`outputSamples`* of `sceAtracSetOutputSamples()`.

Minimum Number of Loop Samples

Minimum number of loop samples

Definition

Value	(Number)	Description
`SCE_ATRAC_AT9_MIN_LOOP_SAMPLES`	3072	Minimum number of loop samples

Description

This identifier indicates the minimum number of loop samples for ATRAC9™.

The number of samples from the loop start sample to the loop end sample must be a value equal to or greater than this identifier.

Infinite Loop Number

Infinite loop number

Definition

Value	(Number)	Description
`SCE_ATRAC_INFINITE_LOOP_NUM`	-1	Infinite loop number

Description

This identifier indicates the number of infinite loops.

Set this identifier when playing back an infinite loop with `sceAtracSetLoopNum()`. In addition, when infinite loop playback is set, this identifier is set to `*`*pLoopNum* of `sceAtracGetLoopInfo()`.

Infinite Sample Number

Infinite sample number

Definition

Value	(Number)	Description
`SCE_ATRAC_INFINITE_SAMPLES`	-1	Infinite sample number

Description

This identifier indicates the number of infinite samples.

Decoder State Identifier

Decoder state identifier

Definition

Value	(Number)	Description
`SCE_ATRAC_DECODER_STATUS_ALL_DATA_WAS_DECODED`	0x00000001	All data is decoded
`SCE_ATRAC_DECODER_STATUS_ALL_DATA_IS_ON_MEMORY`	0x00000002	On-memory playing back
`SCE_ATRAC_DECODER_STATUS_NONLOOP_PART_IS_ON_MEMORY`	0x00000004	Data without a loop or data beyond the loop section is being streaming played back, and all data exists in the buffer.
`SCE_ATRAC_DECODER_STATUS_LOOP_PART_IS_ON_MEMORY`	0x00000008	Data with a loop is being streaming played back, and data required for the currently specified number of loops exists in the buffer.

Description

This identifier indicates the decoder state.

This identifier can be obtained with `sceAtracGetDecoderStatus()`.

Loop State Identifier

Loop state identifier

Definition

Value	(Number)	Description
SCE_ATRAC_LOOP_STATUS_RESETABLE_PART	0x00000001	Can change the number of loops
SCE_ATRAC_LOOP_STATUS_NON_RESETABLE_PART	0x00000000	Cannot change the number of loops

Description

This identifier indicates the loop state.

This identifier can be obtained with `sceAtracGetLoopInfo()`.

Return Codes

List of error codes returned by libatrac

Definition

Value	(Number)	Description
SCE_ATRAC_ERROR_INVALID_POINTER	0x80630000	Invalid pointer argument
SCE_ATRAC_ERROR_INVALID_SIZE	0x80630001	Invalid size
SCE_ATRAC_ERROR_INVALID_WORD_LENGTH	0x80630002	Invalid PCM quantization size
SCE_ATRAC_ERROR_INVALID_TYPE	0x80630003	Invalid ATRAC™ type
SCE_ATRAC_ERROR_INVALID_TOTAL_CH	0x80630004	Invalid total number of channels
SCE_ATRAC_ERROR_INVALID_ALIGNMENT	0x80630005	Invalid alignment
SCE_ATRAC_ERROR_ALREADY_CREATED	0x80630006	Decoder group already created
SCE_ATRAC_ERROR_NOT_CREATED	0x80630007	Decoder group not created
SCE_ATRAC_ERROR_SHORTAGE_OF_CH	0x80630008	Insufficient usable channels
SCE_ATRAC_ERROR_UNSUPPORTED_DATA	0x80630009	Unsupported data
SCE_ATRAC_ERROR_INVALID_DATA	0x8063000A	Invalid data
SCE_ATRAC_ERROR_READ_SIZE_IS_TOO_SMALL	0x8063000B	Set size too small
SCE_ATRAC_ERROR_INVALID_HANDLE	0x8063000C	Invalid handle
SCE_ATRAC_ERROR_READ_SIZE_OVER_BUFFER	0x8063000D	Invalid read size and buffer size
SCE_ATRAC_ERROR_MAIN_BUFFER_SIZE_IS_TOO_SMALL	0x8063000E	Main buffer size too small
SCE_ATRAC_ERROR_SUB_BUFFER_SIZE_IS_TOO_SMALL	0x8063000F	Sub buffer size too small
SCE_ATRAC_ERROR_DATA_SHORTAGE_IN_BUFFER	0x80630010	Insufficient data in buffer
SCE_ATRAC_ERROR_ALL_DATA_WAS_DECODED	0x80630011	All data decoded
SCE_ATRAC_ERROR_INVALID_MAX_OUTPUT_SAMPLES	0x80630012	Invalid number of output samples
SCE_ATRAC_ERROR_ADDED_DATA_IS_TOO_BIG	0x80630013	Invalid size of added data
SCE_ATRAC_ERROR_NEED_SUB_BUFFER	0x80630014	Sub buffer not set
SCE_ATRAC_ERROR_INVALID_SAMPLE	0x80630015	Invalid sample
SCE_ATRAC_ERROR_NO_NEED_SUB_BUFFER	0x80630016	Sub buffer not required
SCE_ATRAC_ERROR_INVALID_LOOP_STATUS	0x80630017	Invalid loop state
SCE_ATRAC_ERROR_REMAIN_VALID_HANDLE	0x80630018	Valid handle remaining
SCE_ATRAC_ERROR_INVALID_LOOP_NUM	0x80630030	Invalid number of loops