Mp4Rec Library Reference

PlayStation®Vita Programmer Tool Runtime Library Release 3.50

Table of Contents

Datatypes

SceMp4RecCtrl

Mp4Rec library control data

Definition

```
#include <mp4rec.h>
typedef struct SceMp4RecCtrl{
        SceUInt32 size;
        void* pHeapBase;
        SceUInt32 heapSize;
        SceInt32 handle;
} SceMp4RecCtrl;
```

Members

size	Size of the structure
pHeapBase	Pointer allocated with `sceKernelGetMemBlockBase()`
heapSize	Size allocated with `sceKernelGetMemBlockBase()`
handle	Handle of the Mp4Rec library instance

Description

This structure is for managing the handle and memory of the Mp4Rec library instance.

Specify `sizeof(SceMp4RecCtrl)` to *size*.

For *pHeapBase* and *heapSize*, set the address and size of the heap memory to use in this API.

For *pHeapBase*, specify the base address obtained with `sceKernelGetMemBlockBase()` after allocating memory with `sceKernelAllocMemBlock()`.

A minimum of `SCE_MP4REC_MIN_HEAP_SIZE` (4 MiB) is required for *heapSize*.

The heap memory is used until `sceMp4RecDeleteRecorder()` is called to delete the Mp4Rec library instance.

For *handle*, the handle of the Mp4Rec library instance will be stored when `sceMp4RecCreateRecorder()` is successful.

See Also

`sceMp4RecCreateRecorder()`, `sceMp4RecDeleteRecorder()`, `sceMp4RecQueryPhysicalMemSize()`, `sceMp4RecInit()`, `sceMp4RecAddVideoSample()`, `sceMp4RecAddAudioSample()`, `sceMp4RecTerm()`

SceMp4RecInitParam

MP4 recorder feature initialization data

Definition

```
#include <mp4rec.h>
typedef struct SceMp4RecInitParam{
        SceUInt32 size;
        SceUInt32 recMode;
        void* pEncoderPhyContPtr;
        SceUInt32 encoderPhyContSize;
        void* pAvRecPhyContPtr;
        SceUInt32 avRecPhyContSize;
        SceInt32 affinityForWriteThread;
        SceInt32 priorityForWriteThread;
} SceMp4RecInitParam;
```

Members

size	Size of the structure
recMode	MP4 recording mode
pEncoderPhyContPtr	Pointer allocated with `sceKernelGetMemBlockBase()`
encoderPhyContSize	AVC encoder work memory size obtained with `sceMp4RecQueryPhysicalMemSize()` (**pEncodeWorkMemSize*)
pAvRecPhyContPtr	Pointer allocated with `sceKernelGetMemBlockBase()`
avRecPhyContSize	AV recorder memory size obtained with `sceMp4RecQueryPhysicalMemSize()` (**pAvRecMemSize*)
affinityForWriteThread	CPU affinity mask for the write thread in the library
priorityForWriteThread	Priority for the write thread in the library

Description

This structure is used to set memory data to initialize the AVC encoder and MP4 recorder feature with `sceMp4RecInit()`. A thread to write to a file is generated in the library. The CPU affinity mask and priority of the thread can be specified in this structure.

For *size*, specify `sizeof(SceMp4RecInitParam)`.

For *recMode*, specify the MP4 recording mode constant. In addition, MP4 recording mode option constants can be specified with a logical OR.

Call `sceMp4RecQueryPhysicalMemSize()` in advance to have the AVC encoder work memory size and AV recorder memory size of the specified *recMode* obtained.

For *pEncoderPhyContPtr*, after using `sceKernelAllocMemBlock()` to allocate an un-cached continuous physical address space (custom DRAM or physically continuous memory on main memory) with 256 KiB alignment in the AVC encoder work memory size, specify the base address that was obtained with `sceKernelGetMemBlockBase()`.

For *encoderPhyContSize*, specify the AVC encoder work memory size obtained with `sceMp4RecQueryPhysicalMemSize()`.

For *pAvRecPhyContPtr*, after using `sceKernelAllocMemBlock()` to allocate an un-cached continuous physical address space (custom DRAM or physically continuous memory on main memory) with 256 byte alignment in the AV recorder memory size, specify the base address that was obtained with `sceKernelGetMemBlockBase()`.

For *`avRecPhyContSize`*, specify the AV recorder memory size obtained with `sceMp4RecQueryPhysicalMemSize()`.

For *`affinityForWriteThread`*, set the CPU affinity mask of the thread to write to a file in the library.

For *`priorityForWriteThread`*, set the priority of the thread to write to a file in the library.

For values of CPU affinity mask and priority, refer to the "Kernel Reference" document.

See Also

`sceMp4RecQueryPhysicalMemSize(),sceMp4RecInit()`

SceMp4RecTermParam

MP4 recorder feature termination data

Definition

```
#include <mp4rec.h>
typedef struct SceMp4RecTermParam{
        SceUInt32 size;
        SceBool isCancel;
        char reserved[512];
        SceMp4RecMetadata* pMetadata;
} SceMp4RecTermParam;
```

Members

size	Size of the structure
isCancel	Cancel flag
reserved	Reserved area (512 bytes)
pMetadata	Pointer to the SceMp4RecMetadata structure

Description

When sceMp4RecTerm() is called, the data recorded up to that point will be output as an MP4 file. To cancel processing after starting recording, set true to *isCancel* and call sceMp4RecTerm(). The MP4 file being recorded will be deleted.

When ending recording to create a MP4 file, set false to *isCancel*.

To set metadata in an MP4 file, specify a pointer to the SceMp4RecMetadata structure in *pMetadata*. Clear with zeroes when not setting metadata.

See Also

sceMp4RecTerm(),SceMp4RecMetadata

SceMp4RecMetadata

Metadata

Definition

```
#include <mp4rec.h>
typedef struct SceMp4RecMetadata{
        SceUInt32 size;
        SceUInt32 lang;
        SceUInt32 titleSize;
        void* pTitle;
        SceUInt32 descriptionSize;
        void* pDescription;
        SceUInt32 copyrightNoticeSize;
        void* pCopyrightNotice;
        SceUInt32 parentalLevel;
        SceUInt32 numOfChapter;
        SceUInt32* pTopOfChapterTbl;
        SceUInt32 numOfThumbnail;
        SceMp4RecThumbnailInfo thumbnailInfo[SCE_MP4REC_THUMBNAIL_NUM_MAX];
} SceMp4RecMetadata;
```

Members

size	Size of the structure
lang	Language of the metadata
titleSize	Size of the title (bytes)
pTitle	Pointer to buffer storing the title
descriptionSize	Size of description (bytes)
pDescription	Pointer to buffer storing description
copyrightNoticeSize	Size of copyright notice (bytes)
pCopyrightNotice	Pointer to buffer storing copyright notice
parentalLevel	Parental control level
numOfChapter	Number of chapters
pTopOfChapterTbl	Pointer to the top of the chapter table
numOfThumbnail	Number of thumbnails (maximum: 2)
thumbnailInfo	Thumbnail information

Description

This structure is for setting metadata to store in an MP4 file.

When setting metadata, `lang` is a required item. Specify the applicable language with the metadata language constant.

Create text metadata for the title, description, and copyright notice in the UTF16-BE format. NULL termination is required at the end of each text. For each metadata size, specify the valid text size including NULL termination. A linefeed character cannot be included in a title.

When not specifying text metadata, make sure sizes and buffer pointers are cleared with zeroes.

The maximum value for `titleSize` is 256 bytes.

The maximum values for `descriptionSize` and `copyrightNoticeSize` are 2000 bytes.

The parental control level specified in *parentalLevel* will be applied to the saved MP4 file; note that it differs from the parental control level of the application. Specify 0 when not setting a parental control level. Regarding the relationship between PlayStation®Vita and the parental control level of contents, refer to "PlayStation®Vita User's Guide".

http://manuals.playstation.net/document/en/psvita/index.html

To set chapters, allocate a continuous area - worth the number of chapters - of the `SceUInt32` type with *pTopOfChapterTbl* as the beginning address and set chapter data to the area in millisecond units. Set the number of chapters in *numOfChapter*. The maximum value for the number of chapters is 999.

When not specifying chapters, clear *numOfChapter* and *pTopOfChapterTbl* with zeroes.

Up to two thumbnails can be set.

The image format that can be used is PNG only.

When specifying a thumbnail, an image of 160 x 120 is required. It is possible to set an image larger than 160 x 120 for the second thumbnail. However, the maximum size for a displayable thumbnail image is 432 x 240. Upon playback, thumbnail of a size larger than 160 x 120 will be prioritized for use.

The maximum size for thumbnail data is 1 MiB.

When not setting thumbnails, specify 0 to *numOfThumbnail*.

See Also

`sceMp4RecTerm()`, `SceMp4RecTermParam`, `SceMp4RecThumbnailInfo`

SceMp4RecThumbnailInfo

Thumbnail information

Definition

```
#include <mp4rec.h>
typedef struct SceMp4RecThumbnailInfo {
        SceUInt32 pngSize;
        void* pPng;
} SceMp4RecThumbnailInfo;
```

Members

pngSize	Data size of thumbnail information
pPng	Pointer to buffer storing thumbnail information

Description

This structure is for setting information of thumbnails in the PNG format.

For *pngSize*, specify the data size of the thumbnail information.

For *pPng*, set the pointer to the buffer storing the thumbnail information.

See Also

`sceMp4RecTerm(),SceMp4RecTermParam,SceMp4RecMetadata`

SceMp4RecFrameOptionRGBA

Frame option RGBA data

Definition

```
#include <mp4rec.h>
typedef struct SceMp4RecFrameOptionRGBA{
        SceUInt8 alpha;
        SceUInt8 cscCoefficient;
        SceUInt8 reserved[14];
} SceMp4RecFrameOptionRGBA;
```

Members

alpha	Specify 0xff
cscCoefficient	Specify the color space conversion coefficient type when converting from RGBA to YCbCr
reserved	Used within library (reserved area)

Description

This structure stores data handled as option data within the picture data.

See Also

`sceMp4RecCsc()`, Color Space Conversion Coefficient Type

SceMp4RecFrameOption

Frame option data

Definition

```
#include <mp4rec.h>
typedef union SceMp4RecFrameOption{
        SceUInt8 reserved[16];
        SceMp4RecFrameOptionRGBA rgba;
} SceMp4RecFrameOption;
```

Members

reserved	Used within library (reserved area)
rgba	Frame option RGBA data

Description

This union stores data handled as option data within the picture data.

Specify *rgba* when using option parameters of RGBA format data with `sceMp4RecCsc()`.

See Also

`sceMp4RecCsc(),SceMp4RecFrameOptionRGBA`

SceMp4RecFrame

Frame data

Definition

```
#include <mp4rec.h>
typedef struct SceMp4RecFrame{
        SceUInt32 pixelType;
        SceUInt32 framePitch;
        SceUInt32 frameWidth;
        SceUInt32 frameHeight;
        SceMp4RecFrameOption opt;

        void *pPicture[2];
} SceMp4RecFrame;
```

Members

pixelType	Pixel type
framePitch	Horizontal frame pitch (in pixels)
frameWidth	Frame width (in pixels)
frameHeight	Frame height (in pixels)
opt	Frame option data
pPicture[0]	Pointer to destination to store frame
pPicture[1]	Reserved area

Description

This structure stores picture data input/output upon encoding with `sceMp4RecCsc()`.

To *pixelType*, specify the constant of the pixel type. However, the input to `sceMp4RecAddVideoSample()` must always be `SCE_MP4REC_PIXEL_YUV420_PACKED_RASTER`. When enabling *opt*, add `SCE_MP4REC_OPTION_ENABLE` as an OR.

To *framePitch*, set the horizontal frame pitch of the area storing the frame data in pixels. Set this in multiples of 16, from 64 to 640 (however, when *pixelType* is `SCE_MP4REC_PIXEL_YUV420_RASTER`, in multiples of 32).

To *frameWidth*, set the frame width of the area storing the frame data in pixels. Set this in multiples of 16, from 64 to 640.

To *frameHeight*, set the frame height of the area storing the frame data in pixels. Set this in multiples of 16, from 64 to 480.

The *framePitch* x *frameHeight* area must be no greater than 640 x 480. Moreover, *frameWidth* must be *framePitch* or less.

Sample input to `sceMp4RecAddVideoSample()` must be of the `SCE_MP4REC_PIXEL_YUV420_PACKED_RASTER` pixel type and *frameWidth* and *frameHeight* must be the same value as the frame size set to the MP4 recording mode of the `SceMp4RecInitParam` structure with `sceMp4RecInit()`. Make sure *framePitch* and *frameWidth* have the same value.

To *pPicture*[0], set the starting address of the area for storing the frame data. Allocate the area for storing the frame data with a 256-byte alignment on un-cached continuous physical address space (custom DRAM or physical continuous memory on main memory).

Because *pPicture*[1] is a reserved area, always assign NULL.

When a change in frame size or pixel type is required, carry out conversion in advance with `sceMp4RecCsc()`.

See Also

`sceMp4RecAddVideoSample()`, `sceMp4RecCsc()`, Pixel Type, `SceMp4RecPicture`, `SceMp4RecFrameOption`

SceMp4RecPicture

Picture data

Definition

```
#include <mp4rec.h>
typedef struct SceMp4RecPicture{
        SceSize size;
        SceMp4RecFrame frame;
} SceMp4RecPicture;
```

Members

size	Size of the structure
frame	Frame data

Description

This structure stores the picture data used when converting the pixel format with `sceMp4RecCsc()`.

Specify `sizeof(SceMp4RecPicture)` to *size*.

See Also

`sceMp4RecAddVideoSample()`, `SceMp4RecFrame`, `SceMp4RecFrameOption`

Library Initialization/Termination

sceMp4RecCreateRecorder

Create Mp4Rec library instance

Definition

```
#include <mp4rec.h>
SceInt32 sceMp4RecCreateRecorder(
        SceMp4RecCtrl *pCtrl
)
```

Arguments

pCtrl Mp4Rec library control data

Return Values

Value	Description
0	Success
Negative number	Error (for details, see "Return Codes")

Description

This function creates an Mp4Rec library instance and sets the heap memory used by the Mp4Rec library.

For *heapSize* of *pCtrl*, set a value larger than SCE_MP4REC_MIN_HEAP_SIZE.
For heap memory allocation, refer to the ”SceMp4RecCtrl” section.

Multiple Mp4Rec instances cannot be created. To create multiple MP4 files, repeat the procedure from creating an Mp4Rec library instance until it is deleted. For basic processing, refer to the ”Mp4Rec Library Overview” document.

Examples

```
SceMp4RecCtrl ctrl;

int res = sceMp4RecCreateRecorder (&ctrl);
```

See Also

sceMp4RecDeleteRecorder()

sceMp4RecDeleteRecorder

Delete the Mp4Rec library instance

Definition

```
#include <mp4rec.h>
SceInt32 sceMp4RecDeleteRecorder(
        SceMp4RecCtrl *pCtrl
)
```

Arguments

pCtrl Mp4Rec library control data

Return Values

Value	Description
0	Success
Negative number	Error (for details, see "Return Codes")

Description

This function deletes the instance of the Mp4Rec library for MP4 recording.

For *pCtrl*, specify a pointer to the SceMp4RecCtrl structure obtained with sceMp4RecCreateRecorder().

Examples

```
SceMp4RecCtrl ctrl;

int res = sceMp4RecDeleteRecorder (&ctrl);
```

See Also

sceMp4RecCreateRecorder()

sceMp4RecQueryPhysicalMemSize

Query physical continuous memory sizes to be used by the AVC encoder and AV recorder

Definition

```
#include <mp4rec.h>
SceInt32 sceMp4RecQueryPhysicalMemSize(
        SceMp4RecCtrl *pCtrl,
        SceUInt32 recMode,
        SceUInt32 *pEncodeWorkMemSize,
        SceUInt32 *pAvRecMemSize
)
```

Arguments

pCtrl	Mp4Rec library control data
recMode	MP4 recording mode
pEncodeWorkMemSize	Pointer to destination to store AVC encoder work memory size
pAvRecMemSize	Pointer to destination to store AV recorder memory size

Return Values

Value	Description
0	Initialization succeeded
Negative number	Error (for details, see "Return Codes")

Description

This function obtains the AVC encoder work memory size and AV recorder memory size required for `sceMp4RecInit()`.

For *pCtrl*, specify the Mp4Rec library control data created with `sceMp4RecCreateRecorder()`.

For *recMode*, specify one of the MP4 recording modes (`SCE_MP4REC_MODE_XXX`).

For *pEncodeWorkMemSize*, the AVC encoder work memory size required for recording will be set.

For *pAvRecMemSize*, the AV recorder memory size required for recording will be set.

Examples

```
SceMp4RecCtrl ctrl;
SceUInt32 encodeWorkMemSize;
SceUInt32 avRecMemSize;
int res = sceMp4RecQueryPhysicalMemSize
        (&ctrl, SCE_MP4REC_MODE_XXX, &encodeWorkMemSize, &avRecMemSize);
```

See Also

`sceMp4RecInit()`

sceMp4RecInit

Initialize the MP4 recorder feature

Definition

```
#include <mp4rec.h>
SceInt32 sceMp4RecInit(
        SceMp4RecCtrl *pCtrl,
        SceMp4RecInitParam *pParam
)
```

Arguments

pCtrl	Mp4Rec library control data
pParam	MP4 recorder feature initialization data

Return Values

Value	Description
0	Initialization succeeded
Negative number	Error (for details, see "Return Codes")

Description

This function initializes the MP4 recorder feature.

For *pCtrl*, specify the Mp4Rec library control data created with `sceMp4RecCreateRecorder()`.

For *pParam*, specify the data for initializing the MP4 recorder feature. For details on the AVC encoder memory and AV recorder memory to set with *pParam*, refer to the ”SceMp4RecInitParam” section.

Examples

```
SceMp4RecCtrl ctrl;
SceUInt32 encodeWorkMemSize;
SceUInt32 avRecMemSize;
int res = sceMp4RecQueryPhysicalMemSize(
        &ctrl, SCE_MP4REC_MODE_XXX, &encodeWorkMemSize, &avRecMemSize);

SceUID encwork = sceKernelAllocMemBlock(
        "encwork",
        SCE_KERNEL_MEMBLOCK_TYPE_USER_MAIN_PHYCONT_NC_RW
        encodeWorkMemSize,0);
if (encwork >= 0) {
        res = sceKernelGetMemBlockBase(
                    encwork, &encPtr);
}

SceUID avRecMem = sceKernelAllocMemBlock(
        "avRecMem",
        SCE_KERNEL_MEMBLOCK_TYPE_USER_MAIN_PHYCONT_NC_RW
        avRecMemSize,0);
if (avRecMem >= 0) {
        res = sceKernelGetMemBlockBase(
                    avRecMem, &avRecPtr);
}

SceMp4RecInitParam param;
```

```
        param.size = sizeof(SceMp4RecInitParam);
        param.recMode = SCE_MP4REC_MODE_XXX;
        param.pEncoderPhyContPtr = encPtr;
        param.encoderPhyContSize = encodeWorkMemSize;
        param.pAvRecPhyContPtr = avRecPtr;
        param.avRecPhyContSize = avRecMemSize;
        param.affinityForWriteThread = SCE_KERNEL_CPU_MASK_USER_ALL;
        param.priorityForWriteThread = SCE_KERNEL_DEFAULT_PRIORITY_USER;

        res = sceMp4RecInit(&ctrl, &param);
```

See Also

`sceMp4RecQueryPhysicalMemSize()`

sceMp4RecTerm

Terminate the MP4 recorder feature

Definition

```
#include <mp4rec.h>
SceInt32 sceMp4RecTerm(
        SceMp4RecCtrl *pCtrl,
        SceMp4RecTermParam *pParam
)
```

Arguments

pCtrl	Mp4Rec library control data
pParam	MP4 recorder feature termination data

Return Values

Value	Description
0	Initialization succeeded
Negative number	Error (for details, see "Return Codes")

Description

This function carries out termination processing of the MP4 recorder feature and makes the created MP4 file usable.

When this function is not called after the end of a recording, the MP4 file is still incomplete and an abnormal file will be created. Make sure to call this function upon ending recording.

For *pCtrl*, specify the Mp4Rec library control data created with `sceMp4RecCreateRecorder()`.

To cancel after starting recording, call this function with `true` specified to *isCancel* of *pParam*. The file being recorded will be deleted.

When setting metadata to an MP4 file, set the pointer to the `SceMp4RecMetadata` structure in *pMetadata* of `SceMp4RecTermParam`. For details on metadata setting, refer to the "SceMp4RecMetadata" section.

Examples

```
SceMp4RecCtrl ctrl;
SceMp4RecTermParam param;
param.size = sizeof(SceMp4RecTermParam);
int res = sceMp4RecTerm(&ctrl, &param);
```

See Also

`sceMp4RecInit()`

MP4 Recording

sceMp4RecAddVideoSample

Add video sample

Definition

```
#include <mp4rec.h>
SceInt32 sceMp4RecAddVideoSample(
        SceMp4RecCtrl *pCtrl,
        void *pBuffer,
        SceInt32 bufSize
)
```

Arguments

pCtrl	Mp4Rec library control data
pBuffer	Pointer to the buffer storing the video sample
bufSize	Size of *pBuffer*

Return Values

Value	Description
0	Success
Negative number	Error (for details, see "Return Codes")

Description

This function adds a video sample (one picture) to the MP4 file.

For *pCtrl*, specify the Mp4Rec library control data created with `sceMp4RecCreateRecorder()`.

For *pBuffer*, specify the video buffer storing the video sample, on which color space conversion was carried out and the pixel type was converted to `SCE_MP4REC_PIXEL_YUV420_PACKED_RASTER`.

For *bufSize*, specify the size of *pBuffer*.

MP4 recording APIs only support encoding of same-size frames. When the original frame buffer size differs with the recording size, size must be aligned. For example, when the frame buffer is 960 x 544 and the MP4 recording mode is `SCE_MP4REC_MODE_640X368_2MBPS`, use `sceMp4RecCsc()` to reduce the frame size to 640 x 368. In this case, the buffer size of one video sample will be 640 x 368 x 1.5 as it is converted to the `SCE_MP4REC_PIXEL_YUV420_PACKED_RASTER` format.

Note that the frame rate is fixed at 29.97 Hz.

Examples

```
SceMp4RecCtrl ctrl;
int res = sceMp4RecAddVideoSample(&ctrl, pBuf, size);
```

See Also

`sceMp4RecAddAudioSample(), sceMp4RecCsc()`

sceMp4RecAddAudioSample

Add audio sample

Definition

```
#include <mp4rec.h>
SceInt32 sceMp4RecAddAudioSample(
        SceMp4RecCtrl *pCtrl,
        void *pBuffer,
        SceInt32 bufSize
)
```

Arguments

pCtrl	Mp4Rec library control data
pBuffer	Pointer to the buffer storing the audio sample
bufSize	Size of *pBuffer*

Return Values

Value	Description
0	Success
Negative number	Error (for details, see "Return Codes")

Description

This function adds an audio sample (1024 samples) to an MP4 file.

For *pCtrl*, specify the Mp4Rec library control data created with `sceMp4RecCreateRecorder()`.

For *pBuffer*, set RAW data of 1024 samples (one frame).

For *bufSize*, specify the size of *pBuffer*. Because encoding specification only allows AAC 16 bit stereo, the size will be 4096 bytes per one frame.

Examples

```
SceMp4RecCtrl ctrl;
int res = sceMp4RecAddAudioSample(&ctrl, pBuf, size);
```

See Also

`sceMp4RecAddVideoSample()`

sceMp4RecCsc

Perform CSC for one picture and expand/reduce frame size

Definition

```
#include <mp4rec.h>
SceInt32 sceMp4RecCsc(
        SceMp4RecPicture *pDst,
        const SceMp4RecPicture *pSrc
)
```

Arguments

pDst	Output picture data
pSrc	Input picture data

Return Values

Value	Description
0	Success
Negative number	Error (for details, see "Return Codes")

Description

This function performs color space conversion for one picture and expands/reduces the frame size so that input can be made to sceMp4RecAddVideoSample().

For *pDst*, specify the buffer storing the parameters to receive after conversion for one picture.

For *pSrc*, input data for one picture before conversion.

Only SCE_MP4REC_PIXEL_YUV420_PACKED_RASTER is supported as the pixel type of a picture that can be specified to sceMp4RecAddVideoSample(). For *pixelType* of *pDst*, specify SCE_MP4REC_PIXEL_YUV420_PACKED_RASTER.

Moreover, the frame size of the picture that can be specified to sceMp4RecAddVideoSample() must be the same value as the frame size of the MP4 recording mode set with sceMp4RecInit(). For *frameWidth* and *frameHeight* of the SceMp4RecFrame structure within the SceMp4RecPicture structure specified to *pDst*, match the frame size with the MP4 recording mode. Make sure *framePitch* and *frameWidth* have the same value.

If the frame size of the input picture of *pSrc* and the output picture of *pDst* differs, the frame size will be expanded/reduced by bilinear interpolation to align to the size of *pDst*. However, note that the expansion/reduction of *frameWidth* and *frameHeight* is limited to 25% - 400%.

When the pixel type of the input picture specified to *pSrc* (*pixelType* of the SceMp4RecFrame structure in the SceMp4RecPicture structure) is SCE_MP4REC_PIXEL_RGBA8888 or SCE_MP4REC_PIXEL_BGRA8888, specify SCE_MP4REC_OPTION_ENABLE in combination to *pixelType* to specify the *cscCoefficient* in the SceMp4RecFrameOptionRGBA structure. This will enable you to specify the coefficient upon color converting from RGBA to YCbCr at the time of picture input.

When specifying *cscCoefficient*, also refer to the "Color Gamut Coefficients (Color Conversion Coefficients)" section in the "Mp4Rec Library Overview" document.

For both *pDst* and *pSrc*, parameters must be set in advance to the picture storing buffer and structures.

For details on the areas of the picture buffers each storing one picture (input/output), refer to the sections of "SceMp4RecPicture" and "SceMp4RecFrame" as well as the "Frame Data of Pictures" and "Restriction on the Buffer for Placing Video Samples" sections of the "Mp4Rec Library Overview" document.

This function can be used even if an Mp4Rec library instance is not created with `sceMp4RecCreateRecorder()` and `sceMp4RecInit()`.

Examples

```
SceMp4RecPicture dst;
SceMp4RecPicture src;

memset(&src,0,sizeof(SceMp4RecPicture));
memset(&dst,0,sizeof(SceMp4RecPicture));

dst.size = src.size = sizeof(SceMp4RecPicture);

src.frame.pixelType = SCE_MP4REC_PIXEL_RGBA8888|SCE_MP4REC_OPTION_ENABLE;
src.frame.framePitch = framebuf.pitch;
src.frame.frameWidth = framebuf.width;
src.frame.frameHeight = framebuf.height;
src.frame.pPicture[0] = framebuf.ptr;
src.frame.opt.rgba.cscCoefficient = SCE_MP4REC_CSC_COEFFICIENT_ITU709;

dst.frame.pixelType = (SceUInt32)SCE_MP4REC_PIXEL_YUV420_PACKED_RASTER;
dst.frame.framePitch = 640;
dst.frame.frameWidth = 640;
dst.frame.frameHeight = 368;
dst.frame.pPicture[0] = ptr;

int res = sceMp4RecCsc(&dst , &src);
```

See Also

`sceMp4RecAddVideoSample()`, `SceMp4RecPicture`, `SceMp4RecFrame`, Pixel Type, Color Space Conversion Coefficient Type

Constants

MP4 Recording Mode

Constant indicating MP4 recording modes

Definition

```
#define SCE_MP4REC_MODE_640X368_2MBPS (0U)
#define SCE_MP4REC_MODE_640X368_1MBPS (1U)
#define SCE_MP4REC_MODE_480X272_2MBPS (2U)
#define SCE_MP4REC_MODE_480X272_1MBPS (3U)
#define SCE_MP4REC_MODE_368X208_2MBPS (4U)
#define SCE_MP4REC_MODE_368X208_1MBPS (5U)
```

Description

This constant indicates MP4 recording modes.

Value	Frame Size	Bit Rate	AVC Level
SCE_MP4REC_MODE_640X368_2MBPS	640 x 368 pixel	2 Mbps	3.0
SCE_MP4REC_MODE_640X368_1MBPS	640 x 368 pixel	1 Mbps	3.0
SCE_MP4REC_MODE_480X272_2MBPS	480 x 272 pixel	2 Mbps	2.1
SCE_MP4REC_MODE_480X272_1MBPS	480 x 272 pixel	1 Mbps	2.1
SCE_MP4REC_MODE_368X208_2MBPS	368 x 208 pixel	2 Mbps	2.1
SCE_MP4REC_MODE_368X208_1MBPS	368 x 208 pixel	1 Mbps	2.1

The following are items common to all modes.

[AVC Video]

Frame Rate	Recording Time Scale	Sample Duration	Profile
29.97 Hz	30000	1001	Baseline

[AAC Audio]

Bit Rate	Recording Time Scale	Sample Duration	Sample Rate
128 Kbps	48000	1024	48 KHz

See Also

`sceMp4RecQueryPhysicalMemSize(),sceMp4RecInit()`

MP4 Recording Mode Options

MP4 recording mode options

Definition

Value	(Number)	Description
SCE_MP4REC_MODE_OPTION_ITU601	0x00010000	ITU-R BT.601 standard
SCE_MP4REC_MODE_OPTION_ITU709	0x00020000	ITU-R BT.709 standard

Description

These are the MP4 recording mode option constants. They can be additionally specified for the MP4 recording mode with a logical OR.

When setting a color gamut coefficient, specify SCE_MP4REC_MODE_OPTION_ITU601 or SCE_MP4REC_MODE_OPTION_ITU709.

See Also

sceMp4RecQueryPhysicalMemSize(),sceMp4RecInit()

Recording Time Scale

Time scale of the MP4 file

Definition

```
#define SCE_MP4REC_VIDEO_TIMESCALE (30000)
#define SCE_MP4REC_AUDIO_TIMESCALE (48000)
```

Description

This is the time scale per one second for recording to an MP4 file.

Sample Duration

Play duration for one frame of the MP4 file

Definition

```
#define SCE_MP4REC_VIDEO_SAMPLE_DURATION (1001)
#define SCE_MP4REC_AUDIO_SAMPLE_DURATION (1024)
```

Description

This is the display duration for one frame when recording to an MP4 file.

The denominator will be the time scale per one second.

Thus, the play duration of one video frame will be 1001/30000 seconds, and the play duration of one audio frame will be 1024/48000 seconds.

Minimum MP4 Recording Time

Minimum time of MP4 recording

Definition

```
#define SCE_MP4REC_DURATION_MIN (1000 * 5) /* msec */
```

Description

This is the minimum time of MP4 recording supported by this library.

A file of less than five seconds cannot be created. When `sceMp4RecTerm()` is called for a file of less than five seconds, dummy data will be inserted within the API to create a five-second MP4 file.

Maximum MP4 Recording Time

Maximum time of MP4 recording

Definition

```
#define SCE_MP4REC_DURATION_MAX (1000 * 60 * 30) /* msec */
```

Description

This is the maximum time of MP4 recording supported by this library.

A file of 30 minutes or longer cannot be created. When samples are added exceeded 30 minutes and `sceMp4RecAddVideoSample()` or `sceMp4RecAddAudioSample()` is called, `SCE_MP4REC_ERROR_DURATION_LIMIT` will return. Terminate recording with `sceMp4RecTerm()`.

Minimum Heap Memory Size

Minimum size of heap memory

Definition

```
#define SCE_MP4REC_MIN_HEAP_SIZE (1024*1024*4)
```

Description

This is the minimum value for *`heapSize`* of the `SceMp4RecCtrl` structure that is specified to `sceMp4RecCreateRecorder()`.

This library requires a heap size of at least 4 MiB.

Pixel Type

Constant indicating the pixel type

Definition

```
#define SCE_MP4REC_PIXEL_RGBA8888                 (0x0U)
#define SCE_MP4REC_PIXEL_BGRA8888                 (0x8U)
#define SCE_MP4REC_PIXEL_YUV420_RASTER            (0x10U)
#define SCE_MP4REC_PIXEL_YUV420_PACKED_RASTER     (0x20U)
#define SCE_MP4REC_OPTION_ENABLE                  (1<<31)
```

Description

This constant indicates the pixel type.

Set this to the *pixelType* member of the SceMp4RecFrame structure.

When specifying the *opt* member of the SceMp4RecFrame structure, also specify SCE_MP4REC_OPTION_ENABLE.

When used in sceMp4RecAddVideoSample(), the pixel type must be SCE_MP4REC_PIXEL_YUV420_PACKED_RASTER.

See Also

SceMp4RecFrame, sceMp4RecAddVideoSample(), sceMp4RecCsc()

Color Space Conversion Coefficient Type

Constant indicating the color conversion coefficient for picture input in color space conversion

Definition

Value	(Number)	Description
`SCE_MP4REC_CSC_COEFFICIENT_DEFAULT`	0	Default color gamut coefficient (ITU-R BT.601 standard)
`SCE_MP4REC_CSC_COEFFICIENT_ITU601`	1	ITU-R BT.601 standard
`SCE_MP4REC_CSC_COEFFICIENT_ITU709`	2	ITU-R BT.709 standard

Description

This constant indicates the coefficient when color converting from RGBA to YCbCr upon picture input with `sceMp4RecCsc()`.

Specify this to *`cscCoefficient`* of the `SceMp4RecFrameOptionRGBA` structure, which is the *`rgba`* member in the `SceMp4RecFrameOption` union, which in turn is the *`opt`* member of the `SceMp4RecFrame` structure.

When using this constant, also specify `SCE_MP4REC_OPTION_ENABLE` to the *`pixelType`* member of the `SceMp4RecFrame` structure.

See Also

`SceMp4RecFrameOptionRGBA`, `SceMp4RecFrame`, `sceMp4RecCsc()`

Metadata Language

Constants for representing metadata language

Definition

```
#define SCE_MP4REC_LANG_JAPANESE    (0x2a0e)
#define SCE_MP4REC_LANG_ENGLISH     (0x15c7)
#define SCE_MP4REC_LANG_FRENCH      (0x1a41)
#define SCE_MP4REC_LANG_SPANISH     (0x4e01)
#define SCE_MP4REC_LANG_GERMAN      (0x10b5)
#define SCE_MP4REC_LANG_ITALIAN     (0x2681)
#define SCE_MP4REC_LANG_DUTCH       (0x3984)
#define SCE_MP4REC_LANG_PORTUGUESE  (0x41f2)
#define SCE_MP4REC_LANG_RUSSIAN     (0x4ab3)
#define SCE_MP4REC_LANG_KOREAN      (0x2df2)
#define SCE_MP4REC_LANG_CHINESE     (0x690f)
#define SCE_MP4REC_LANG_FINNISH     (0x192e)
#define SCE_MP4REC_LANG_SWEDISH     (0x4ee5)
#define SCE_MP4REC_LANG_DANISH      (0x102e)
#define SCE_MP4REC_LANG_NORWEGIAN   (0x39f2)
#define SCE_MP4REC_LANG_POLISH      (0x41ec)
#define SCE_MP4REC_LANG_TURKISH     (0x52b2)
```

Description

These constants represent languages in which the metadata is written. The specification of the metadata language using this constant is required when setting metadata.

See Also

`sceMp4RecTerm(),SceMp4RecMetadata`

Thumbnail Frame Size

Frame sizes of a thumbnail to specify in metadata

Definition

```
#define SCE_MP4REC_THUMBNAIL_WIDTH_REQUIRED       (160)
#define SCE_MP4REC_THUMBNAIL_HEIGHT_REQUIRED      (120)
#define SCE_MP4REC_THUMBNAIL_WIDTH_MAX            (432)
#define SCE_MP4REC_THUMBNAIL_HEIGHT_MAX           (240)
```

Description

These are frame size constants for thumbnails of metadata.

When specifying a thumbnail in metadata, an image of 160 x 120 is required.

An image of size larger than 160 x 120 can be specified as an option.

See Also

`sceMp4RecTerm(),SceMp4RecMetadata,SceMp4RecThumbnailInfo`

Return Codes

List of return codes returned by the Mp4Rec library

Definition

Value	(Number)	Description
`SCE_MP4REC_ERROR_INVALID_ARG`	0x80108301	Invalid argument
`SCE_MP4REC_ERROR_OUT_OF_MEMORY`	0x80108302	Not enough memory
`SCE_MP4REC_ERROR_INVALID_RECMODE`	0x80108303	Invalid MP4 recording mode
`SCE_MP4REC_ERROR_INVALID_BUFSIZE`	0x80108304	Invalid buffer size
`SCE_MP4REC_ERROR_NOT_INITIALIZED`	0x80108305	Not initialized
`SCE_MP4REC_ERROR_DURATION_LIMIT`	0x80108306	Exceeded recording limit
`SCE_MP4REC_ERROR_INTERNAL`	0x80108307	Internal error
`SCE_MP4REC_ERROR_NOSPACE`	0x80108308	Not enough storage space
`SCE_MP4REC_ERROR_STATE`	0x80108309	Invalid status
`SCE_MP4REC_ERROR_VIDEOENC_INVALID_PARAM`	0x80108342	Invalid video encoder argument
`SCE_MP4REC_ERROR_VIDEOENC_OUT_OF_MEMORY`	0x80108343	Not enough memory in the video encoder
`SCE_MP4REC_ERROR_VIDEOENC_INVALID_STATE`	0x80108344	Invalid video encoder state
`SCE_MP4REC_ERROR_VIDEOENC_UNSUPPORT_IMAGE_SIZE`	0x80108345	Image size not supported by the video encoder
`SCE_MP4REC_ERROR_VIDEOENC_INVALID_COLOR_FORMAT`	0x80108346	Invalid color format
`SCE_MP4REC_ERROR_VIDEOENC_NOT_PHY_CONTINUOUS_MEMORY`	0x80108347	Not physical continuous memory area
`SCE_MP4REC_ERROR_VIDEOENC_ALREADY_USED`	0x80108348	Video encoder is already being used
`SCE_MP4REC_ERROR_VIDEOENC_INVALID_POINTER`	0x80108349	Invalid pointer
`SCE_MP4REC_ERROR_VIDEOENC_INITIALIZE`	0x8010834B	Error occurred during video encoder initialization
`SCE_MP4REC_ERROR_VIDEOENC_NOT_INITIALIZE`	0x8010834C	Video encoder is not initialized
`SCE_MP4REC_ERROR_VIDEOENC_INVALID_ARGUMENT_SIZE`	0x8010834E	Value of `sizeof` (structure) is not specified in *`size`* member of structure
`SCE_MP4REC_ERROR_AVCENC_INVALID_PARAM`	0x80108352	Invalid AVC encoder argument
`SCE_MP4REC_ERROR_AVCENC_OUT_OF_MEMORY`	0x80108353	Not enough memory in the AVC encoder
`SCE_MP4REC_ERROR_AVCENC_INVALID_STATE`	0x80108354	Invalid AVC encoder state
`SCE_MP4REC_ERROR_AVCENC_UNSUPPORT_IMAGE_SIZE`	0x80108355	Image size not supported by the AVC encoder
`SCE_MP4REC_ERROR_AVCENC_INVALID_COLOR_FORMAT`	0x80108356	Invalid color format
`SCE_MP4REC_ERROR_AVCENC_NOT_PHY_CONTINUOUS_MEMORY`	0x80108357	Not physical continuous memory area
`SCE_MP4REC_ERROR_AVCENC_ALREADY_USED`	0x80108358	AVC encoder is already being used
`SCE_MP4REC_ERROR_AVCENC_INVALID_POINTER`	0x80108359	Invalid pointer

Value	(Number)	Description
SCE_MP4REC_ERROR_ AVCENC_PICTURE_BUFFER_FULL	0x8010835A	Picture buffer of the AVC encoder overflowed
SCE_MP4REC_ERROR_AVCENC_INITIALIZE	0x8010835B	Error occurred during AVC encoder initialization
SCE_MP4REC_ERROR_AVCENC_NOT_INITIALIZE	0x8010835C	AVC encoder is not initialized
SCE_MP4REC_ERROR_ AVCENC_INVALID_ENCODE_PARAM	0x8010835D	Invalid AVC encoder argument
SCE_MP4REC_ERROR_ AVCENC_INVALID_ARGUMENT_SIZE	0x8010835E	Value of `sizeof` (structure) is not specified in *`size`* member of structure
SCE_MP4REC_ERROR_AVCENC_MAXAUSIZE	0x80108361	Exceeded maximum AU size
SCE_MP4REC_ERROR_AACENC_INVALIDARG	0x80108381	Invalid AAC encoder argument
SCE_MP4REC_ERROR_AACENC_INVALIDHANDLE	0x80108382	Invalid AAC encoder handle
SCE_MP4REC_ERROR_AACENC_OUTOFMEMORY	0x80108383	Not enough memory in the AAC encoder
SCE_MP4REC_ERROR_AACENC_INTERNAL	0x80108384	AAC encoder internal error
SCE_MP4REC_ERROR_AACENC_FRAME_LIMIT	0x80108385	Exceeded maximum frame size of the AAC encoder
SCE_MP4REC_ERROR_AACENC_NOTIMPL	0x80108386	Unimplemented feature was called
SCE_MP4REC_ERROR_AVREC_OUT_OF_MEMORY	0x801083A1	Not enough memory in the AV recorder
SCE_MP4REC_ERROR_AVREC_FATAL	0x801083A2	Fatal error in AV recorder
SCE_MP4REC_ERROR_AVREC_INVALID_VALUE	0x801083A3	Invalid AV recorder argument
SCE_MP4REC_ERROR_AVREC_VENC_BUFFER_FULL	0x801083A4	Encode buffer of the AV recorder overflowed
SCE_MP4REC_ERROR_AVREC_FILE_ACCESS	0x801083A5	AV recorder cannot access file
SCE_MP4REC_ERROR_AVREC_FILE_OPERATION	0x801083A6	AV recorder cannot operate file
SCE_MP4REC_ERROR_AVREC_NOT_IMPLEMENT	0x801083A7	Unimplemented feature was called
SCE_MP4REC_ERROR_AVREC_BUFFER_FULL	0x801083A8	Buffer overflowed in the AV recorder
SCE_MP4REC_ERROR_ AVREC_SAMPLE_NUM_BUFFER_FULL	0x801083A9	Exceeded maximum multiplex buffering samples of the AV recorder
SCE_MP4REC_ERROR_AVREC_VIDEO_BUFFER_FULL	0x801083AA	Multiplex buffer of the AV recorder overflowed
SCE_MP4REC_ERROR_ AVREC_VIDEO_SAMPLE_NUM_BUFFER_FULL	0x801083AB	Exceeded maximum video multiplex buffering samples of the AV recorder
SCE_MP4REC_ERROR_AVREC_AUDIO_BUFFER_FULL	0x801083AC	Audio multiplex buffer of the AV recorder overflowed
SCE_MP4REC_ERROR_ AVREC_AUDIO_SAMPLE_NUM_BUFFER_FULL	0x801083AD	Exceeded maximum audio multiplex buffering samples of the AV recorder
SCE_MP4REC_ERROR_ AVREC_VENC_RET_INVALID_POINTER	0x801083AE	Invalid pointer

Value	(Number)	Description
SCE_MP4REC_ERROR_AVREC_NAACENC_INVALIDARG	0x801083B1	Invalid AAC encoder argument
SCE_MP4REC_ERROR_ AVREC_NAACENC_INVALIDHANDLE	0x801083B2	Invalid AAC encoder handle
SCE_MP4REC_ERROR_ AVREC_NAACENC_OUTOFMEMORY	0x801083B3	Not enough memory in the AAC encoder
SCE_MP4REC_ERROR_AVREC_NAACENC_INTERNAL	0x801083B4	Internal error
SCE_MP4REC_ERROR_ AVREC_NAACENC_FRAME_LIMIT	0x801083B5	Exceeded maximum frame size of the AAC encoder
SCE_MP4REC_ERROR_AVREC_NAACENC_NOTIMPL	0x801083B6	Unimplemented feature was called
SCE_MP4REC_ERROR_AVREC_NAACENC_FATAL	0x801083B7	Fatal error in AAC encoder
SCE_MP4REC_ERROR_AVREC_FF4_ARGS	0x801083C1	Invalid MP4 multiplexer argument
SCE_MP4REC_ERROR_AVREC_FF4_STATE	0x801083C2	Invalid MP4 multiplexer state
SCE_MP4REC_ERROR_AVREC_FF4_NOMEM	0x801083C3	Not enough memory in the MP4 multiplexer
SCE_MP4REC_ERROR_AVREC_FF4_FILE	0x801083C4	Invalid file access in MP4 multiplexer
SCE_MP4REC_ERROR_AVREC_FF4_NOTFOUND	0x801083C5	File cannot be found in MP4 multiplexer
SCE_MP4REC_ERROR_AVREC_FF4_OVERFLOW	0x801083C6	Buffer overflow in MP4 multiplexer
SCE_MP4REC_ERROR_AVREC_FF4_FORMAT	0x801083C7	Invalid format in MP4 multiplexer
SCE_MP4REC_ERROR_AVREC_FF4_INTERNAL	0x801083C8	Internal error in the MP4 multiplexer
SCE_MP4REC_ERROR_AVREC_FF4_FATAL	0x801083C9	Fata l error in MP4 multiplexer
SCE_MP4REC_ERROR_EXPORT_PARAM	0x801083E1	Invalid export argument
SCE_MP4REC_ERROR_EXPORT_FILE_NOT_FOUND	0x801083E2	Export file cannot be found
SCE_MP4REC_ERROR_EXPORT_CONTENT_FULL	0x801083E3	Exceeded maximum number of contents that can be registered
SCE_MP4REC_ERROR_EXPORT_NO_MEMORY	0x801083E4	Ran out of memory during export
SCE_MP4REC_ERROR_EXPORT_SERVER_DOWN	0x801083E5	Server is not running
SCE_MP4REC_ERROR_EXPORT_TOO_MANY_CLIENT	0x801083E6	Exceeded maximum server connections
SCE_MP4REC_ERROR_EXPORT_MEDIA_FULL	0x801083E7	Not enough storage space
SCE_MP4REC_ERROR_EXPORT_CREATE_FILE	0x801083E8	Temporary file creation error
SCE_MP4REC_ERROR_ EXPORT_NOT_SUPPORTED_FORMAT	0x801083E9	Invalid file format
SCE_MP4REC_ERROR_EXPORT_CANCELED	0x801083EA	Export was canceled
SCE_MP4REC_ERROR_EXPORT_MEDIA_NOT_EXIST	0x801083EB	No memory card
SCE_MP4REC_ERROR_EXPORT_DB_CORRUPTED	0x801083EC	Database is corrupted
SCE_MP4REC_ERROR_EXPORT_FILE_EXIST	0x801083ED	Export target file already exists
SCE_MP4REC_ERROR_EXPORT_INTERNAL	0x801083EE	Internal export error
SCE_MP4REC_ERROR_EXPORT_NOT_INITIALIZED	0x801083EF	Export is not initialized

Value	(Number)	Description
`SCE_MP4REC_ERROR_EXPORT_OPEN_FAILED`	0x801083F0	Export destination target device does not exist
`SCE_MP4REC_ERROR_EXPORT_NOT_ENOUGH_AREA`	0x801083F1	Not enough area to export
`SCE_MP4REC_ERROR_EXPORT_NOT_VALID_STR`	0x801083F2	String of export argument is not valid
`SCE_MP4REC_ERROR_` `EXPORT_NOT_VALID_FILENAME`	0x801083F3	Export filename is not valid