

JPEG Decoder Reference

© 2012 Sony Computer Entertainment Inc.
All Rights Reserved.
SCE Confidential

Table of Contents

Motion JPEG Decode Function.....	3
sceJpegInitMjpeg	4
sceJpegInitMjpegWithParam	5
sceJpegDecodeMjpeg	6
sceJpegDecodeMjpegYCbCr.....	8
sceJpegMjpegCsc.....	10
sceJpegFinishMjpeg	12
SceJpegMjpegInitParam	13
Obtaining JPEG Information	14
sceJpegGetOutputInfo	15
SceJpegOutputInfo	17
Color Space Conversion Functions	19
sceJpegCsc.....	20
Split Decoding	22
sceJpegCreateSplitDecoder	23
sceJpegSplitDecodeMjpeg.....	24
sceJpegDeleteSplitDecoder	26
SceJpegSplitDecodeCtrl	27
Constants	29
Color Space Type.....	30
Color Space Sampling.....	31
Return Codes	32

Motion JPEG Decode Function

SCE CONFIDENTIAL

sceJpegInitMjpeg

Initialize Motion JPEG Decoder

Definition

```
#include <scejpeg.h>
int sceJpegInitMjpeg(
    int maxSplitDecoder
)
```

Arguments

maxSplitDecoder Maximum number of Split Decoders to be used simultaneously (0 to 8)

Return Values

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

Description

This function initializes Motion JPEG decoder.

To *maxSplitDecoder*, specify the maximum number of Split Decoders to be used simultaneously. Specify 0 if the split decoding feature will not be used.

This function is initialized to access only the video memory as input/output buffer region to pass data to decoder. In order to use the main memory, use `sceJpegInitMjpegWithParam()` instead of this function.

Notes

This function is not multithread safe.

Example

```
int res = sceJpegInitMjpeg(0);
```

See Also

`sceJpegFinishMjpeg()`

SCE CONFIDENTIAL

sceJpegInitMjpegWithParam

Initialize Motion JPEG Decoder (parameter-specified version)

Definition

```
#include <scejpeg.h>
int sceJpegInitMjpegWithParam(
    const SceJpegMjpegInitParam *pInitParam
)
```

Arguments

pInitParam parameter to be used at initialization

Return Values

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

Description

This function initializes Motion JPEG decoder.

For *pInitParam*, specify parameters to be used at initialization.

Notes

This function is not multithread safe.

Examples

```
int res;
SceJpegMjpegInitParam initParam;

initParam.size      = sizeof(SceJpegMjpegInitParam);
initParam.maxSplitDecoder = 0;
initParam.option    = SCE_JPEG_MJPEG_INIT_OPTION_NONE;

ret = sceJpegInitMjpegWithParam(&initParam);
```

See Also

SceJpegMjpegInitParam, sceJpegFinishMjpeg()

SCE CONFIDENTIAL

sceJpegDecodeMJpeg

Decode and rapidly convert color space of Motion JPEG data

Definition

```
#include <scejpeg.h>
int sceJpegDecodeMJpeg (
    const unsigned char *pJpeg,
    SceSize isize,
    void *pRGBA,
    SceSize osize,
    int decodeMode,
    void *pTempBuffer,
    SceSize tempBufferSize,
    void *pCoefBuffer,
    SceSize coefBufferSize
)
```

Arguments

<i>pJpeg</i>	Pointer to JPEG data to be decoded
<i>isize</i>	JPEG data size (in bytes)
<i>pRGBA</i>	Pointer to output buffer
<i>osize</i>	Output buffer size (in bytes)
<i>decodeMode</i>	Parameter related to decoding method
<i>pTempBuffer</i>	Pointer to color space conversion working buffer
<i>tempBufferSize</i>	Size of color space conversion working buffer (in bytes)
<i>pCoefBuffer</i>	Pointer to quantization coefficient buffer
<i>coefBufferSize</i>	Quantization coefficient buffer size (in bytes)

Return Values

Value	Description
>0	Output data pixel count
< 0	Error (for details, see "Return Codes")

Description

This function decodes JPEG data and stores it in RGBA8888 format in the specified area.

If the function completes normally, a value equal to (number of pixels in the x direction of the decoded data)* 65536 + (number of pixels in the y direction of the decoded data) will be returned. The output size (size after downscaling when downscaling) of supported JPEG data using YCbCr422/420 format is limited to the range between 64 x 64 pixels and 2032 x 1088 pixels.

To *pJpeg* and *isize*, specify the JPEG data to be decoded.

To *pRGBA* and *osize*, specify the buffer to store the decoded result data. The output data size will depend on the JPEG image size and the specification of *decodeMode* (explained below). Note that the specified number of pixels processed in the Decoder will be rounded to the nearest MCU unit (a multiple of 16 for YCbCr420).

To *decodeMode*, specify the parameter related to the decoding method. Select a value from the following table that indicates the processing method of the DHT segment and color space constraints. Add option values as necessary and specify the total value.

SCE CONFIDENTIAL

Value	Description
SCE_JPEG_MJPEG_WITH_DHT	Decodes JPEG data containing a DHT segment (limited to YCbCr422/420)
SCE_JPEG_MJPEG_WITHOUT_DHT	Decodes JPEG data without a DHT segment using the JPEG standard recommended Huffman table (limited to YCbCr422/420, for Motion JPEG)

The following options can be specified.

Downscale Magnification

To downscale the output image size, add one of the following values. When using this feature, the sizes of the decoded result output buffer (*pRGB* and *osize*) and the color space conversion working buffer (*pTempBuffer* and *tempBufferSize*) should be calculated based on the reduced image size.

Value	Description
SCE_JPEG_MJPEG_DOWNSCALE_1_2	Reduce image size to 1/2
SCE_JPEG_MJPEG_DOWNSCALE_1_4	Reduce image size to 1/4
SCE_JPEG_MJPEG_DOWNSCALE_1_8	Reduce image size to 1/8

To *pTempBuffer* and *tempBufferSize*, specify the working buffer where the decoded JPEG data in YCbCr will be temporarily stored. Round the JPEG image size to be decoded in MCU units (a multiple of 16 for YCbCr420). The pitch difference required by the color space conversion hardware (round the size of each line for Y, Cb, Cr to a multiple of 16) must be added to this size.

To *pCoefBuffer* and *coefBufferSize*, specify the quantization coefficient buffer that will be used to decode the progressive JPEG. If the JPEG data to be decoded is not a progressive JPEG, specify NULL to *pCoefBuffer*, and 0 to *coefBufferSize*. When decoding progressive JPEG data, first allocate the memory for the quantization coefficient buffer size obtained with `sceJpegGetOutputInfo()`, then specify this memory.

Notes

This function is multithread safe.

The buffer specifying the JPEG data (*pJpeg*), output buffer (*pRGBA*), color space conversion working buffer (*pTempBuffer*), and quantization coefficient buffer (*pCoefBuffer*) have several restrictions. Refer to "Buffer Restrictions for Positioning I/O Data" in "Precautions" under the "JPEG Decoder Overview" document for details.

By using `sceJpegGetOutputInfo()`, the memory size required for the output buffer, color space conversion working buffer and quantization coefficient buffer can be obtained easily.

Example

```
int res;

res = sceJpegDecodeMjpeg(
    jpegData, 100*1024,
    framebuf, 320*240*4,
    SCE_JPEG_MJPEG_WITH_DHT,
    tempbuf, 320*240*3/2, /* for YCbCr420 */
    NULL, 0 /* for baseline JPEG */ );
```

See Also

`sceJpegGetOutputInfo()`

SCE CONFIDENTIAL

sceJpegDecodeMjpegYCbCr

Decode Motion JPEG data

Definition

```
#include <scejpeg.h>
int sceJpegDecodeMjpegYCbCr (
    const unsigned char *pJpeg,
    SceSize isize,
    unsigned char *pYCbCr,
    SceSize osize,
    int decodeMode,
    void *pCoefBuffer,
    SceSize coefBufferSize
)
```

Arguments

<i>pJpeg</i>	Pointer to JPEG data to be decoded
<i>isize</i>	JPEG data size (in bytes)
<i>pYCbCr</i>	Pointer to output buffer
<i>osize</i>	Output buffer size (in bytes)
<i>decodeMode</i>	Parameter related to decoding method
<i>pCoefBuffer</i>	Pointer to quantization coefficient buffer
<i>coefBufferSize</i>	Quantization coefficient buffer size (in bytes)

Return Values

Value	Description
>0	Output data pixel count
< 0	Error (for details, see "Return Codes")

Description

This function decodes JPEG data and stores the results in a specified area.

If the function completes normally, a value equal to (number of pixels in the x direction of the decoded data)* 65536 + (number of pixels in the y direction of the decoded data) will be returned

To *pJpeg* and *isize*, specify the JPEG data to be decoded.

To *pYCbCr* and *osize*, specify the buffer for storing the decoded result data. The output data size will depend on the JPEG image size and the specification of *decodeMode* (explained below). Note that the specified number of pixels processed within the decoder will be rounded to the nearest MCU unit (a multiple of 16 for YCbCr420). Also note that when `SCE_JPEG_MJPEG_WITH_DHT` or `SCE_JPEG_MJPEG_WITHOUT_DHT` is specified to *decodeMode*, the pitch difference required by the color space conversion hardware (the size of each line for Y, Cb, Cr is rounded to a multiple of 16) will be added to this number.

To *decodeMode*, specify the parameters related to the decoding method. Select a value from the following table that indicates the processing method of the DHT segment and color space constraints. Add any option values as necessary, and specify the total value.

SCE CONFIDENTIAL

Value	Description
SCE_JPEG_MJPEG_WITH_DHT	Decodes JPEG data with a DHT segment (limited to YCbCr422/420)
SCE_JPEG_MJPEG_WITHOUT_DHT	Decodes JPEG data without a DHT segment using the JPEG standard recommended Huffman table (limited to YCbCr422/420, for Motion JPEG)
SCE_JPEG_MJPEG_ANY_SAMPLING_WITHOUT_DHT	Decodes JPEG data without a DHT segment using the JPEG standard recommended Huffman table (no color space constraints, for Motion JPEG)
SCE_JPEG_MJPEG_ANY_SAMPLING	Decodes general JPEG data (no color space constraints)

The following options can be specified.

Downscale Magnification

To downscale the output image size, add one of the following values. When using this feature, calculate the output buffer size of the decoded result (*pYCbCr* and *osize*) based on the reduced image size.

Value	Description
SCE_JPEG_MJPEG_DOWNSCALE_1_2	Reduce image size to 1/2
SCE_JPEG_MJPEG_DOWNSCALE_1_4	Reduce image size to 1/4
SCE_JPEG_MJPEG_DOWNSCALE_1_8	Reduce image size to 1/8

To *pCoefBuffer* and *coefBufferSize*, specify the quantization coefficient buffer that will be used to decode the progressive JPEG. If the JPEG data to be decoded is not a progressive JPEG, specify NULL to *pCoefBuffer*, and 0 to *coefBufferSize*. When decoding progressive JPEG data, first allocate the memory for the quantization coefficient buffer size obtained using `sceJpegGetOutputInfo()`, then specify this memory.

Notes

This function is multithread safe.

The buffer specifying the JPEG data (*pJpeg*), output buffer (*pRGBA*), and quantization coefficient buffer (*pCoefBuffer*) have several restrictions. Refer to "Buffer Restrictions for Positioning I/O Data" in "Precautions" under the "JPEG Decoder Overview" document for details.

By using `sceJpegGetOutputInfo()`, the memory size required for the output buffer and quantization coefficient buffer can be obtained easily.

Example

```
int res;

res = sceJpegDecodeMjpegYCbCr(
    jpegData, 100*1024,
    ycbcr, 320*240*3/2, /* for YCbCr420 */
    SCE_JPEG_MJPEG_WITH_DHT,
    NULL, 0 /* for baseline JPEG */ );
```

See Also

`sceJpegGetOutputInfo()`

sceJpegMjpegCsc

Convert color space of Motion JPEG decoded result

Definition

```
#include <scejpeg.h>
int sceJpegMjpegCsc (
    void *pRGBA,
    const unsigned char *pYCbCr,
    int xysize,
    int iFrameWidth,
    int colorOption,
    int sampling
)
```

Arguments

<i>pRGBA</i>	Pointer to frame buffer where color space conversion result is stored
<i>pYCbCr</i>	Pointer to input buffer where YCbCr data is stored
<i>xysize</i>	Horizontal and vertical pixel count of input buffer
<i>iFrameWidth</i>	Width of output frame buffer (in pixels)
<i>colorOption</i>	Output data format and input data color space
<i>sampling</i>	Color space sampling and conversion method of input data

Return Values

Value	Description
0	Processing completed normally
< 0	Error (for details, see "Return Codes")

Description

This function performs color space conversion on the decoded result obtained from `sceJpegDecodeMjpegYCbCr()` to RGBA format.

Supported input images are limited to the result within the range of 64 x 64 pixels to 2032 x 1088 pixels in YCbCr422/420 format decoded with `SCE_JPEG_MJPEG_WITH_DHT` / `SCE_JPEG_MJPEG_WITHOUT_DHT` specified to the argument *decodeMode* of the `sceJpegDecodeMjpegYCbCr()` function. If this function is used for any other images, its behavior is not guaranteed.

To *pRGBA*, specify the pointer to the frame buffer where the color space conversion results are stored. The output buffer size will not be checked, so specify sufficient memory area to store the converted image.

To *pYCbCr* and *xysize*, specify the YCbCr image data to be performed color space conversion. To *pYCbCr*, specify the pointer to the decoded result buffer passed to the `sceJpegDecodeMjpegYCbCr()` function, and pass the return value of the same function to *xysize*.

To *iFrameWidth*, specify the width of the output frame buffer in pixels. The value can be specified in multiples of 4, ranging from the width of the input image to 2032.

To *colorOption*, specify the output data format and input data color space. For the output format, select one of the following values.

Value	Description
<code>SCE_JPEG_PIXEL_RGBA8888</code>	Output in RGBA8888 format

SCE CONFIDENTIAL

Value	Description
SCE_JPEG_PIXEL_BGRA8888	Output in BGRA8888 format

The input data color space can be specified by adding the following values to the values listed above.

Value	Description
SCE_JPEG_COLORSPACE_JFIF	Color space defined by JFIF (default)
SCE_JPEG_COLORSPACE_BT601	Color space defined by BT.601

To *sampling*, specify the color space sampling and conversion method of the input data.

Use the following formula to calculate the color space sampling of the input data.

- (Vertical color space sampling coefficient) * 256 + (horizontal color space sampling coefficient)

This color space sampling can use the last 16 bits of the return value from `sceJpegGetOutputInfo()`, or a constant indicating the type of color space sampling (defined separately).

Notes

This function is multithread safe.

The input / output buffers (`pRGBA` and `pYCbCr`) have several restrictions. Refer to "Buffer Restrictions for Positioning I/O Data" in "Precautions" under the "JPEG Decoder Overview" document for details.

Example

```
int res;

res = sceJpegDecodeMjpegYCbCr(
    jpegData, 100*1024,
    ycbcr, 320*240*3/2, /* for YCbCr420 */
    SCE_JPEG_MJPEG_WITH_DHT,
    NULL, 0 /* for baseline JPEG */ );
if (res < 0) {
    /* Error handling */
}
res = sceJpegMjpegCsc(
    framebuf, ycbcr, res, 320,
    SCE_JPEG_PIXEL_RGBA8888 | SCE_JPEG_COLORSPACE_JFIF,
    SCE_JPEG_CS_H2V2);
```

See Also

`sceJpegDecodeMjpegYCbCr()`

SCE CONFIDENTIAL

sceJpegFinishMjpeg

Terminate Motion JPEG Decoder

Definition

```
#include <scejpeg.h>
int sceJpegFinishMjpeg(
    void
)
```

Arguments

None

Return Values

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

Description

This function terminates Motion JPEG decoding.

Notes

This function is not multithread safe.

Example

```
int res = sceJpegFinishMjpeg();
```

See Also

sceJpegInitMjpeg()

SCE CONFIDENTIAL

SceJpegMjpegInitParam

Initialization parameter of Motion JPEG Decoder

Definition

```
#include <scejpeg.h>
typedef struct {
    SceSize size;
    int maxSplitDecoder;
    int option;
} SceJpegMjpegInitParam;
```

Members

<i>size</i>	Number of bytes of this structure (sizeof(SceJpegMjpegInitParam))
<i>maxSplitDecoder</i>	Maximum number of Split Decoders to be used simultaneously (0 to 8)
<i>option</i>	Option

Description

This structure is used to store parameters to pass when initializing Motion JPEG decode with `sceJpegInitMjpegWithParam()`.

To *size*, specify the size of this structure in bytes.

To *maxSplitDecoder*, specify the maximum number of Split Decoders to be used simultaneously. Specify 0 if the split decoding feature will not be used.

To *option*, specify an initialization option. Specify `SCE_JPEG_MJPEG_INIT_OPTION_NONE` to utilize video memory as an input/output buffer region to pass data to the decoder. If you wish to use only the main memory, or both video/main memories, use `SCE_JPEG_MJPEG_INIT_OPTION_LPDDR2_MEMORY` instead.

See Also

`sceJpegInitMjpegWithParam()`

Obtaining JPEG Information

SCE CONFIDENTIAL

sceJpegGetOutputInfo

Obtain JPEG Decoder output information

Definition

```
#include <scejpeg.h>
int sceJpegGetOutputInfo (
    const unsigned char *pJpeg,
    SceSize isize,
    int outputFormat,
    int decodeMode,
    SceJpegOutputInfo *pOutputInfo
)
```

Arguments

<i>pJpeg</i>	Pointer to JPEG data to be decoded
<i>isize</i>	JPEG data size (in bytes)
<i>outputFormat</i>	Output data format
<i>decodeMode</i>	Parameter related to decoding method
<i>pOutputInfo</i>	Pointer to where obtained information is stored

Return Values

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

Description

This function obtains the size of each buffer type, color space information and pitch information, all of which are required to decode the specified JPEG data.

To *pJpeg* and *isize*, specify the JPEG data containing the required information.

To *outputFormat*, specify the output data format. When `sceJpegDecodeMjpeg()` is used for output in RGBA format, specify `SCE_JPEG_PIXEL_RGBA8888` or `SCE_JPEG_PIXEL_BGRA8888`, and when `sceJpegDecodeMjpegYCbCr()` is used for output without color space conversion, specify `SCE_JPEG_NO_CSC_OUTPUT`.

To *decodeMode*, specify the parameters related to the decoding method. Specify the value to be specified to `sceJpegDecodeMjpeg()` or `sceJpegDecodeMjpegYCbCr()`. Refer to the description of each function for details.

The obtained information is stored in *pOutputInfo*.

Notes

This function is multithread safe.

The buffer specifying the JPEG data (*pJpeg*) has several restrictions. Refer to "Buffer Restrictions for Positioning I/O Data" in "Precautions" under the "JPEG Decoder Overview" document for details.

SCE CONFIDENTIAL

Example

```
int res;
SceJpegOutputInfo outputInfo;

res = sceJpegGetOutputInfo(
    jpegData, 100*1024,
    SCE_JPEG_PIXEL_RGBA8888,
    SCE_JPEG_MJPEG_WITH_DHT,
    &outputInfo);
```

See Also

SceJpegOutputInfo, sceJpegDecodeMjpeg(), sceJpegDecodeMjpegYCbCr()

SceJpegOutputInfo

JPEG Decoder output information

Definition

```
#include <scejpeg.h>
typedef struct {
    int colorSpace;
    unsigned short imageWidth;
    unsigned short imageHeight;
    SceSize outputBufferSize;
    SceSize tempBufferSize;
    SceSize coefBufferSize;
    struct {
        unsigned int x;
        unsigned int y;
    } pitch[4];
} SceJpegOutputInfo;
```

Members

<i>colorSpace</i>	Color space information
<i>imageWidth</i>	Image width (in pixels)
<i>imageHeight</i>	Image height (in lines)
<i>outputBufferSize</i>	Required size for output buffer (in bytes)
<i>tempBufferSize</i>	Required size for color space conversion working buffer (in bytes)
<i>coefBufferSize</i>	Required size for quantization coefficient buffer (in bytes)
<i>pitch</i>	Pitch information (in bytes/line)

Description

This structure is used to store information obtained from JPEG data using `sceJpegGetOutputInfo()`.

colorSpace contains the total value of the constant indicating the color space type and the constant indicating the color space sampling. Refer to the description of each constant for details.

imageWidth and *imageHeight* contain the image size. This is the image size if decoding is performed at equal magnification without downscaling.

outputBufferSize contains the required output buffer size. If `SCE_JPEG_PIXEL_RGBA8888` is specified using the *outputFormat* argument in `sceJpegGetOutputInfo()`, this is the size of the color space converted RGBA data. If -1 is specified, this is the data size of the color space in the JPEG data.

tempBufferSize contains the size required for the color space conversion working buffer.

coefBufferSize contains the size required for the quantization coefficient buffer. If the JPEG file is not progressive, the quantization coefficient buffer is unused, so the value will be 0.

pitch contains the pitch size for the x direction and y direction. If `SCE_JPEG_PIXEL_RGBA8888` is specified in the *outputFormat* argument of `sceJpegGetOutputInfo()`, *pitch[0]* will contain pitch information in RGBA. If -1 is specified, *pitch[0]* will contain Y pitch information, *pitch[1]* will contain Cb pitch information, and *pitch[2]* will contain Cr pitch information. Concerning the starting addresses for Cb and Cr data, the starting address for Cb is *pitch[0].x* x *pitch[0].y* added to the head address of the output buffer. And the starting address for Cr is *pitch[1].x* x *pitch[1].y* added to the head address of Cb.

SCE CONFIDENTIAL

See Also

sceJpegGetOutputInfo ()

000004892117

Color Space Conversion Functions

sceJpegCsc

Color space conversion from YCbCr444 format

Definition

```
#include <scejpeg.h>
int sceJpegCsc(
    void *pRGBA,
    const unsigned char *pYCbCr,
    int xysize,
    int iFrameWidth,
    int colorOption,
    int sampling
)
```

Arguments

<i>pRGBA</i>	Pointer to frame buffer where color space conversion result is stored
<i>pYCbCr</i>	Pointer to input buffer where YCbCr data is stored
<i>xyysize</i>	Horizontal and vertical pixel count of input buffer
<i>iFrameWidth</i>	Width of output frame buffer (in pixels)
<i>colorOption</i>	Output data format and input data color space
<i>sampling</i>	Color space sampling and conversion method of input data

Return Values

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

Description

This function performs color space conversion on the decoded result obtained from `sceJpegDecodeMjpegYCbCr()` to RGBA format.

Supported input images are limited to the YCbCr444 format data decoded with the `sceJpegDecodeMjpegYCbCr()` function with specification of `SCE_JPEG_MJPEG_ANY_SAMPLING` or `SCE_JPEG_MJPEG_ANY_SAMPLING_WITHOUT_DHT` to the argument *decodeMode*. While no restriction is implemented on the pixel count of an input image or the width of output frame buffer, color space is limited to `SCE_JPEG_COLORSPACE_JFIF`, and output format is limited to `SCE_JPEG_PIXEL_RGBA8888`, and speed may be slow.

Specification method of each argument is similar to `sceJpegMjpegCsc()`. For details, refer to the description of `sceJpegMjpegCsc()`.

Notes

This function is multithread safe.

The input / output buffers (*pRGBA* and *pYCbCr*) have several restrictions. Refer to "Buffer Restrictions for Positioning I/O Data" in "Precautions" under the "JPEG Decoder Overview" document for details.

SCE CONFIDENTIAL

Examples

```
int res;

res = sceJpegDecodeMjpegYCbCr(
    jpegData, 100*1024,
    ycbcr, 320*240*3, /* for YCbCr444 */
    SCE_JPEG_MJPEG_ANY_SAMPLING,
    NULL, 0 /* for baseline JPEG */ );
if (res < 0) {
    /* Error handling */
}
res = sceJpegCsc(
    framebuf, ycbcr, res, 320,
    SCE_JPEG_PIXEL_RGBA8888 | SCE_JPEG_COLORSPACE_JFIF,
    SCE_JPEG_CS_H1V1);
```

See Also

```
sceJpegMjpegCsc(), sceJpegDecodeMjpegYCbCr()
```

Split Decoding

000004892117

sceJpegCreateSplitDecoder

Generate Split Decoder

Definition

```
#include <scejpeg.h>
int sceJpegCreateSplitDecoder(
    SceJpegSplitDecodeCtrl *pCtrl
)
```

Arguments

pCtrl Storage location for split decoding control information

Return Values

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

Description

This function generates the Split Decoder.

To *pCtrl*, specify the storage location of the split decoding control information. Specify the allocated memory area by the user side. This control information is used to distinguish the Split Decoder instance until the Split Decoder is discarded.

The maximum number of Split Decoders that can be used simultaneously can be specified using the *maxSplitDecoder* argument in *sceJpegInitMjpeg()*. If the specified maximum number is reached, this function will return *SCE_JPEG_ERROR_EXCEED_MAX_SPLIT_DECODER*.

Notes

When this function uses a separate split decoder instance for each thread, it is multithread safe.

Example

```
SceJpegSplitDecodeCtrl ctrl;

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

See Also

SceJpegSplitDecodeCtrl, *sceJpegDeleteSplitDecoder()*

SCE CONFIDENTIAL

sceJpegSplitDecodeMjpeg

Process split decoding

Definition

```
#include <scejpeg.h>
int sceJpegSplitDecodeMjpeg (
    SceJpegSplitDecodeCtrl *pCtrl
)
```

Arguments

pCtrl Split decoding control information

Return Values

Refer to the description.

Description

JPEG data for 1 frame is split and entered into the Decoder several times, and the results are stored in the specified area.

To *pCtrl*, specify the split decoding control information obtained using `sceJpegCreateSplitDecoder()`. At the first call, the stream buffer (*pStreamBuffer* and *streamBufferSize*) and decode mode (*decodeMode*) must be specified to the control information, and the head section of the JPEG data to be decoded must be loaded into the stream buffer.

Although this function advances the decoding by loading the JPEG data in strips and repeatedly calling it, depending on the return value, appropriate processing may be required on the application side.

Return Values	Description
SCE_JPEG_ERROR_INIT_DONE	Indicates completion of the JPEG data header decoding. At this point, the decode output information is stored in the control information member <i>outputInfo</i> . Use this information to allocate the output buffer and quantization coefficient buffer, and specify them to the members <i>pOutputBuffer</i> and <i>pCoefBuffer</i> . (If the JPEG data to be decoded is not progressive, the quantization coefficient buffer is not required) Then, call this function again to continue decoding.
SCE_JPEG_ERROR_INPUT_SUSPENDED	Indicates completion of the decoding of the JPEG data section loaded in the stream buffer. Load only the <i>writeBufferSize</i> of successive JPEG data to the address indicated by <i>pWriteBuffer</i> in the control information. If the remaining JPEG data size is equal to or less than <i>writeBufferSize</i> , specify the actual size loaded to <i>writeBufferSize</i> , and 1 to <i>isEndOfStream</i> . Then, call this function again to continue decoding.
0 or higher	Indicates the normal completion of the decoding. This value is the output pixel count expressed in the following format: (number of pixels in the x direction)*65536+(number of pixels in the y direction). Then discard the Split Decoder.
Other values	Indicates that the split decoding cannot continue due to an error. Perform the required error handling and discard the Split Decoder.

SCE CONFIDENTIAL

Notes

When this function uses a separate split decoder instance for each thread, it is multithread safe.

Example

```
SceJpegSplitDecodeCtrl ctrl;

int res = sceJpegCreateSplitDecoder(&ctrl);
if (res < 0) {
    /* Error handling */
}

ctrl.pStreamBuffer = streamBuffer;
ctrl.streamBufferSize = /* Specify the number of bytes in streamBuffer */;
ctrl.decodeMode = SCE_JPEG_MJPEG_WITH_DHT;
/* Load stream to ctrl.streamBuffer */

for (;;) {
    res = sceJpegSplitDecodeMjpeg(&ctrl);
    if (res == SCE_JPEG_ERROR_INPUT_SUSPENDED) {
        /* Load stream to ctrl.pWriteBuffer */
    } else if (res == SCE_JPEG_ERROR_INIT_DONE) {
        /*Specify output buffer and quantization coefficient buffer */
        ctrl.pOutputBuffer = ybcr;
    } else if (res >= 0) {
        /* Decoding complete */
        break;
    } else {
        /* Error handling */
        break;
    }
}
```

See Also

SceJpegSplitDecodeCtrl

SCE CONFIDENTIAL

sceJpegDeleteSplitDecoder

Discard Split Decoder

Definition

```
#include <scejpeg.h>
int sceJpegDeleteSplitDecoder(
    SceJpegSplitDecodeCtrl *pCtrl
)
```

Arguments

pCtrl Split decoding control information

Return Values

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

Description

This function discards the Split Decoder.

To *pCtrl*, specify the Split Decoder control information to be discarded.

Notes

When this function uses a separate split decoder instance for each thread, it is multithread safe.

Example

```
SceJpegSplitDecodeCtrl ctrl;

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

See Also

SceJpegSplitDecodeCtrl, sceJpegCreateSplitDecoder()

SceJpegSplitDecodeCtrl

Split decoding control information

Definition

```
#include <scejpeg.h>
typedef struct {
    unsigned char *pStreamBuffer;
    SceSize streamBufferSize;
    unsigned char *pWriteBuffer;
    SceSize writeBufferSize;
    int isEndOfStream;
    int decodeMode;
    SceJpegOutputInfo outputInfo;
    void *pOutputBuffer;
    void *pCoefBuffer;
    unsigned int internalData[3];
} SceJpegSplitDecodeCtrl;
```

Members

<i>pStreamBuffer</i>	Pointer to stream buffer
<i>streamBufferSize</i>	Stream buffer size
<i>pWriteBuffer</i>	Pointer to write buffer
<i>writeBufferSize</i>	Write buffer size
<i>isEndOfStream</i>	If data written this time is end of stream
<i>decodeMode</i>	Parameter related to decoding method
<i>outputInfo</i>	Decoding output information
<i>pOutputBuffer</i>	Pointer to output buffer
<i>pCoefBuffer</i>	Pointer to quantization coefficient buffer
<i>internalData</i>	Area to be used within library

Description

This information is used to control the split decoding.

To *pStreamBuffer* and *streamBufferSize*, specify the start address and size of the stream buffer. Before calling `sceJpegSplitDecodeMjpeg()` for the first time, specify the memory allocated by the application. Once this memory is specified, the address and size cannot be changed until the Split Decoder is discarded. Be sure to specify the buffer size of the stream separated and input multiple times in multiples of 256.

By calling `sceJpegSplitDecodeMjpeg()`, information about the buffer where subsequent JPEG data can be written is stored in *pWriteBuffer* and *writeBufferSize*. This area always indicates the stream buffer range.

To *isEndOfStream*, specify if the data written this time to *pWriteBuffer* is the end of the stream. If the written data includes the end of the stream, specify 1. Otherwise, specify 0.

To *decodeMode*, specify the parameters related to the decoding method. The value that can be specified will be the same as *decodeMode* in `sceJpegDecodeMjpegYCbCr()`. Refer to the description of `sceJpegDecodeMjpegYCbCr()`.

The decoded output information is stored in *outputInfo*. This will become valid when `SCE_JPEG_ERROR_INIT_DONE` is returned by calling `sceJpegSplitDecodeMjpeg()` repeatedly.

SCE CONFIDENTIAL

To *pOutputBuffer*, specify the buffer where data from the decoding results are stored. Repeatedly call `sceJpegSplitDecodeMjpeg()` until `SCE_JPEG_ERROR_INIT_DONE` is returned. Then specify this before calling `sceJpegSplitDecodeMjpeg()`. The required buffer size is stored in *outputInfo.outputBufferSize*. Allocate this memory size and specify its head address.

To *pCoefBuffer*, specify the pointer to the quantization coefficient buffer that will be used for decoding the progressive JPEG. Repeatedly call `sceJpegSplitDecodeMjpeg()` until `SCE_JPEG_ERROR_INIT_DONE` is returned. Then specify this before calling `sceJpegSplitDecodeMjpeg()`. The required buffer size is stored in *outputInfo.coefBufferSize*. Allocate this memory size and specify its head address. If *outputInfo.coefBufferSize* is 0, the JPEG data is not progressive, so the buffer does not need to be specified.

Notes

Each type of buffers (*pStreamBuffer*, *pOutputBuffer*, and *pCoefBuffer*) has several restrictions. Refer to "Buffer Restrictions for Positioning I/O Data" in "Precautions" under the "JPEG Decoder Overview" document for details.

See Also

`sceJpegCreateSplitDecoder()`, `sceJpegDeleteSplitDecoder()`,
`sceJpegSplitDecodeMjpeg()`

Constants

000004892117

SCE CONFIDENTIAL

Color Space Type

Constants indicating color space type

Definition

```
#define SCE_JPEG_CS_UNKNOWN      (0<<16)      /* unknown */
#define SCE_JPEG_CS_GRAYSCALE   (1<<16)      /* Y only */
#define SCE_JPEG_CS_YCBCR       (2<<16)      /* Y/Cb/Cr */
```

Description

These constants indicate the color space type.

See Also

SceJpegOutputInfo, sceJpegMjpegCsc ()

000004892117

Color Space Sampling

Constants indicating color space sampling types

Definition

```
#define SCE_JPEG_CS_H1V1    (0x101)    /* 1:1 */
#define SCE_JPEG_CS_H2V1    (0x201)    /* 2:1 */
#define SCE_JPEG_CS_H4V1    (0x401)    /* 4:1 */
#define SCE_JPEG_CS_H1V2    (0x102)    /* 1:2 */
#define SCE_JPEG_CS_H2V2    (0x202)    /* 2:2 */
#define SCE_JPEG_CS_H1V4    (0x104)    /* 1:4 */
```

Description

These constants indicate the types of color space sampling.

The top 8 bits correspond to the color space sampling coefficient in the vertical direction, and the bottom 8 bits correspond to the color space sampling coefficient in the horizontal direction.

Currently, all other color space sampling is not supported.

See Also

SceJpegOutputInfo, sceJpegMjpegCsc()

SCE CONFIDENTIAL

Return Codes

List of JPEG Decoder Return Codes

Definition

Value	Hexadecimal	Description
SCE_OK	0	Normal completion
SCE_JPEG_ERROR_IMAGE_EMPTY	0x80650003	Image size is 0
SCE_JPEG_ERROR_BAD_MARKER_LENGTH	0x80650004	Invalid size marker found
SCE_JPEG_ERROR_BAD_DHT_COUNTS	0x80650005	Number of Huffman tables is invalid
SCE_JPEG_ERROR_BAD_DHT_INDEX	0x80650006	A non-existent Huffman table has been specified
SCE_JPEG_ERROR_BAD_DQT_INDEX	0x80650007	A non-existent inverse quantization segment has been specified
SCE_JPEG_ERROR_DECODE_ERROR	0x80650009	Variable-length code decoding error
SCE_JPEG_ERROR_INVALID_POINTER	0x80650010	Invalid pointer argument
SCE_JPEG_ERROR_BAD_COMPONENT_ID	0x80650011	A non-existent color dimension has been specified
SCE_JPEG_ERROR_UNSupport_COLORSPACE	0x80650013	An unsupported color space has been specified
SCE_JPEG_ERROR_BAD_MCU_SIZE	0x80650014	Invalid MCU size
SCE_JPEG_ERROR_BAD_PRECISION	0x80650015	Invalid DCT coefficient precision
SCE_JPEG_ERROR_UNSupport_SAMPLING	0x80650016	Unsupported color space sampling
SCE_JPEG_ERROR_COMPONENT_COUNT	0x80650017	Number of color space dimensions is invalid
SCE_JPEG_ERROR_EOI_EXPECTED	0x80650019	No EOI was found
SCE_JPEG_ERROR_UNSupport_IMAGE_SIZE	0x80650020	Image size is too big
SCE_JPEG_ERROR_NO_HUFF_TABLE	0x80650021	An undefined Huffman table has been used
SCE_JPEG_ERROR_NO_QUANT_TABLE	0x80650022	An undefined inverse quantization coefficient has been used
SCE_JPEG_ERROR_NO_SOI	0x80650023	No SOI maker was found
SCE_JPEG_ERROR_BAD_DQT_MARKER	0x80650024	Invalid DQT marker found
SCE_JPEG_ERROR_BAD_DHT_MARKER	0x80650025	Invalid DHT marker found
SCE_JPEG_ERROR_BAD_DRI_MARKER	0x80650026	Invalid DRI marker found
SCE_JPEG_ERROR_BAD_SOF_MARKER	0x80650027	Invalid SOF marker found
SCE_JPEG_ERROR_BAD_SOS_MARKER	0x80650028	Invalid SOS marker found
SCE_JPEG_ERROR_SOF_DUPLICATE	0x80650029	Multiple SOF markers exist
SCE_JPEG_ERROR_NO_LOSSLESS_SUPPORT	0x80650031	Lossless compression is not supported
SCE_JPEG_ERROR_NO_ARITH_SUPPORT	0x80650032	Arithmetic compression is not supported
SCE_JPEG_ERROR_UNKNOWN_MARKER	0x80650035	Unknown marker found
SCE_JPEG_ERROR_UNEXPECTED_MARKER	0x80650037	Unexpected marker found
SCE_JPEG_ERROR_INVALID_REGION	0x80650038	An invalid region was specified
SCE_JPEG_ERROR_INVALID_STATE	0x80650039	A function was called from an inappropriate state
SCE_JPEG_ERROR_MEMORY_SIZE	0x80650041	Invalid memory size was specified
SCE_JPEG_ERROR_CANNOT_INIT	0x80650042	Cannot initialize
SCE_JPEG_ERROR_CANNOT_FINISH	0x80650043	Cannot finish
SCE_JPEG_ERROR_INVALID_COLOR_FORMAT	0x80650050	Color space format is not supported by color space conversion feature

SCE CONFIDENTIAL

Value	Hexadecimal	Description
SCE_JPEG_ERROR_NOT_PHY_CONTINUOUS_MEMORY	0x80650053	Physical address of specified memory area is not continuous
SCE_JPEG_ERROR_INVALID_DECODE_MODE	0x80650060	Invalid decode mode was specified
SCE_JPEG_ERROR_BAD_PROGRESSIVE_PARAM	0x80650061	Invalid progressive parameter
SCE_JPEG_ERROR_EXCEED_MAX_SPLIT_DECODER	0x80650062	Maximum number of Split Decoders generated for simultaneous use has been exceeded
SCE_JPEG_ERROR_INIT_DONE	0x80650063	Initialization of Split Decoder has finished
SCE_JPEG_ERROR_INPUT_SUSPENDED	0x80650064	Split decoding processing has been suspended
SCE_JPEG_ERROR_INPUT_DATA_TOO_BIG	0x80650065	Input data size has exceeded stream buffer size
SCE_JPEG_ERROR_INVALID_DATA_SIZE	0x80650066	Specified data size is too big
SCE_JPEG_ERROR_INVALID_INIT_PARAM	0x80650067	Invalid value was specified to the initialization parameter