AVC Decoder Reference

PlayStation®Vita Programmer Tool Runtime Library Release 3.50

Document serial number: 000004892117

Document serial number: 000004892117

Table of Contents

Datatypes

SceVideodecQueryInitInfoHwAvcdec

Initialization parameter

Definition

```
#include <videodec.h>
typedef struct SceVideodecQueryInitInfoHwAvcdec{
        SceSize size;
        SceUInt32 horizontal;
        SceUInt32 vertical;
        SceUInt32 numOfRefFrames;
        SceUInt32 numOfStreams;
} SceVideodecQueryInitInfoHwAvcdec;
```

Members

size	Specify `sizeof(SceVideodecQueryInitInfoHwAvcdec)`
horizontal	Maximum width of the decode image (in pixels)
vertical	Maximum height of the decode image (in pixels)
numOfRefFrames	Maximum number of reference images upon decoding
numOfStreams	Maximum number of AVC decoders to be used simultaneously (1)

Description

This structure is used to specify the parameters when initializing the library with `sceVideodecInitLibrary()`.

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

To *horizontal*, specify the maximum width of the stream image to input to the AVC decoder. Specify this in units of 16 pixels from 64 to 1920.

To *vertical*, specify the maximum height of the stream image to input to the AVC decoder. Specify this in units of 16 pixels from 64 to 1088.

In addition, the *horizontal* x *vertical* area must be no greater than 1280 x 720.

To *numOfRefFrames*, specify the maximum number of reference images of the stream to be input to the AVC decoder with the maximum number of Lv 3.1. (Imprudently increasing this number will cause an increase in used memory. The recommended value for normal streams is 3.)

To *numOfStreams*, specify the maximum number of AVC decoders to be used simultaneously. Specify 1.

See Also

`sceVideodecInitLibrary()`

SceVideodecQueryInitInfo

Initialization parameter

Definition

```
#include <videodec.h>
typedef union SceVideodecQueryInitInfo{
        SceUInt8 reserved[32];
        SceVideodecQueryInitInfoHwAvcdec hwAvc;
} SceVideodecQueryInitInfo;
```

Members

hwAvc Initialization parameter of AVC decoder

Description

This union is used to specify the parameter when initializing the library with `sceVideodecInitLibrary()`.

When using an AVC decoder, set the parameter to *hwAvc*.

See Also

`sceVideodecInitLibrary(),SceVideodecQueryInitInfoHwAvcdec`

SceVideodecMemInfo

Video decoder memory information

Definition

```
#include <videodec.h>
typedef struct SceVideodecMemInfo{
        SceUInt32 memSize;
} SceVideodecMemInfo;
```

Members

memSize Number of bytes of memory required to generate video decoder instances

Description

This structure handles the memory size used by the video decoder that is required to perform library initialization with `sceVideodecInitLibraryWithUnmapMem()`.

The required memory size can be obtained with `sceVideodecQueryMemSize()`, so allocate the required memory size with 256 KiB alignment in an uncached continuous physical address space (custom DRAM or physical continuous memory on the main memory) by using `sceCodecEngineAllocMemoryFromUnmapMemBlock()`. Then assign the start pointer and the allocated size to *vaContext* and *contextSize* of the `SceVideodecCtrl` structure respectively and call `sceVideodecInitLibraryWithUnmapMem()`.

See Also

`sceVideodecInitLibraryWithUnmapMem()`, `sceVideodecQueryMemSize()`, `sceCodecEngineAllocMemoryFromUnmapMemBlock()`, `SceVideodecCtrl`

SceVideodecBuf

Video decoder buffer parameter

Definition

```
#include <videodec.h>
typedef struct SceVideodecBuf{
        void *pBuf;
        SceUInt32 size;
} SceVideodecBuf;
```

Members

pBuf Starting address of buffer
size Buffer size

Description

This structure is used to indicate the buffer with the video decoder.

To *pBuf*, assign the starting address of the corresponding buffer.

To *size*, assign the size of the corresponding buffer.

See Also

SceVideodecCtrl

SceVideodecCtrl

Video decoder memory size information

Definition

```
#include <videodec.h>
typedef struct SceVideodecCtrl{
        SceVideodecBuf memBuf;
        SceUID memBufUid;
        SceUIntVAddr vaContext;
        SceUInt32 contextSize;
} SceVideodecCtrl;
```

Members

memBuf	Always specify 0
memBufUid	Always specify 0
vaContext	Pointer allocated with `sceCodecEngineAllocMemoryFromUnmapMemBlock()`
contextSize	Size allocated with `sceCodecEngineAllocMemoryFromUnmapMemBlock()`

Description

This structure handles the memory size used by the video decoder that is required to perform library initialization with `sceVideodecInitLibraryWithUnmapMem()`.

The required memory size can be obtained with `sceVideodecQueryMemSize()`, so allocate the required memory size with 256 KiB alignment in an uncached continuous physical address space (custom DRAM or physical continuous memory on the main memory) by using `sceCodecEngineAllocMemoryFromUnmapMemBlock()`. Then assign the start pointer and the allocated size to *vaContext* and *contextSize* of the `SceVideodecCtrl` structure respectively and call `sceVideodecInitLibraryWithUnmapMem()`.

See Also

`sceVideodecInitLibraryWithUnmapMem()`, `sceVideodecQueryMemSize()`, `sceCodecEngineAllocMemoryFromUnmapMemBlock()`

SceAvcdecQueryDecoderInfo

AVC decoder memory information

Definition

```
#include <videodec.h>
typedef struct SceAvcdecQueryDecoderInfo{
        SceUInt32 horizontal;
        SceUInt32 vertical;
        SceUInt32 numOfRefFrames;
} SceAvcdecQueryDecoderInfo;
```

Members

horizontal	Maximum width of the decode image (in pixels)
vertical	Maximum height of the decode image (in pixels)
numOfRefFrames	Maximum number of reference images upon decoding

Description

This structure is used to obtain the frame memory size used by the AVC decoder that is required to create an AVC decoder instance with `sceAvcdecQueryDecoderMemSize()` and `sceAvcdecCreateDecoder()`.

To *horizontal*, specify the maximum width of the stream image to input to the AVC decoder. Specify this in units of 16 pixels from 64 to 1920.

To *vertical*, specify the maximum height of the stream image to input to the AVC decoder. Specify this in units of 16 pixels from 64 to 1088.

In addition, the *horizontal* x *vertical* area must be no greater than 1280 x 720.

To *numOfRefFrames*, specify the maximum number of images of the stream to be input to the AVC decoder with the maximum number of Lv 3.1. (Imprudently increasing this number will cause an increase in used memory. The recommended value for normal streams is 3.)

See Also

`sceAvcdecQueryDecoderMemSize()`, `sceAvcdecCreateDecoder()`

SceAvcdecDecoderInfo

AVC decoder memory size information

Definition

```
#include <videodec.h>
typedef struct SceAvcdecDecoderInfo{
        SceUInt32 frameMemSize;
} SceAvcdecDecoderInfo;
```

Members

frameMemSize Number of frame memory bytes required to create an AVC decoder instance

Description

This structure is used to handle the frame memory size used by the AVC decoder that is required to create an AVC decoder instance with `sceAvcdecCreateDecoder()`.

The required memory size can be obtained with `sceAvcdecQueryDecoderMemSize()`, so allocate the required memory size with 1-MiB alignment in an uncached continuous physical address space (custom DRAM or physical continuous memory on the main memory), and then assign the start pointer and allocated size to *frameBuf* of the `SceAvcdecCtrl` structure and call `sceAvcdecCreateDecoder()`.

See Also

`sceAvcdecQueryDecoderMemSize()`, `sceAvcdecCreateDecoder()`

SceAvcdecCtrl

AVC decoder memory size information

Definition

```
#include <videodec.h>
typedef struct SceAvcdecCtrl{
        SceUInt32 handle;
        SceAvcdecBuf frameBuf;
} SceAvcdecCtrl;
```

Members

handle	Handle to the AVC decoder instance
frameBuf	Specify the pointer and number of bytes of the frame memory required to create an AVC decoder instance

Description

When `sceAvcdecCreateDecoder()` completes successfully, the handle to the AVC decoder instance is stored to *handle*.

The required memory size can be obtained with `sceAvcdecQueryDecoderMemSize()` to *frameBuf*, so allocate the required memory size in an uncached continuous physical address space (custom DRAM or physical continuous memory on the main memory), and then assign the start pointer and allocated size to *pBuf* and *size*, respectively, and call `sceAvcdecCreateDecoder()`.

See Also

`sceAvcdecQueryDecoderMemSize()`, `sceAvcdecCreateDecoder()`

SceAvcdecBuf

AVC decoder buffer parameter

Definition

```
#include <videodec.h>
typedef struct SceAvcdecBuf{
        void *pBuf;
        SceUInt32 size;
} SceAvcdecBuf;
```

Members

pBuf Starting address of buffer
size Buffer size

Description

This structure is used to indicate the buffer with the AVC Decoder library.

To *pBuf*, assign the starting address of the corresponding buffer.

To *size*, assign the size of the corresponding buffer.

See Also

SceAvcdecCtrl, SceAvcdecAu

SceVideodecTimeStamp

AVC decoder time stamp

Definition

```
#include <videodec.h>
typedef struct SceVideodecTimeStamp{
        SceUInt32 upper;
        SceUInt32 lower;
} SceVideodecTimeStamp;
```

Members

upper High-order 32 bits of the time stamp
lower Low-order 32 bits of the time stamp

Description

This structure is used to indicate the time with the AVC Decoder library.

To *upper*, assign the high-order 32 bits of the time. To *lower*, assign the low-order 32 bits of the time.

Store them in units of 90 kHz.

If the value is invalid, assign `SCE_VIDEODEC_VOID_TIMESTAMP(0xffffffff)` to both the *upper* and *lower* member variables.

Refer to "Time Stamp of Input Access Unit" in "AVC Decoder Overview" for details.

See Also

`SceAvcdecAu`

SceAvcdecAu

AVC decoder access unit

Definition

```
#include <videodec.h>
typedef struct SceAvcdecAu{
        SceVideodecTimeStamp pts;
        SceVideodecTimeStamp dts;
        SceAvcdecBuf es;
} SceAvcdecAu;
```

Members

pts	Display time of access unit
dts	Decoding time of access unit
es	Buffer storing one access unit

Description

This structure is used to indicate the access unit ("AU") that is input when decoding with `sceAvcdecDecode()` in the AVC Decoder library.

Assign the AU display time to *pts*, and assign the AU decoding time to *dts*. Refer to "Time Stamp of Input Access Unit" in "AVC Decoder Overview" for details.

To *es*, specify the buffer to which one AU is stored.

See Also

`sceAvcdecDecode()`

SceAvcdecInfo

Optional decoding information after AVC decoding

Definition

```
#include <videodec.h>
typedef struct SceAvcdecInfo{
        SceUInt32 numUnitsInTick;
        SceUInt32 timeScale;
        SceUChar8 fixedFrameRateFlag;

        SceUChar8 aspectRatioIdc;
        SceUShort16 sarWidth;
        SceUShort16 sarHeight;

        SceUChar8 colourPrimaries;
        SceUChar8 transferCharacteristics;
        SceUChar8 matrixCoefficients;

        SceUChar8 videoFullRangeFlag;

        SceUChar8 reserved[3];

        SceUChar8 flag;
        SceUChar8 padding[2];
        SceVideodecTimeStamp outputPts;

} SceAvcdecInfo;
```

Members

numUnitsInTick	Same value as `num_units_in_tick` of AVC standard
timeScale	Same value as `time_scale` of AVC standard
fixedFrameRateFlag	Same value as `fixed_frame_rate_flag` of AVC standard
aspectRatioIdc	Same value as `aspect_ratio_idc` of AVC standard
sarWidth	Same value as `sar_width` of AVC standard
sarHeight	Same value as `sar_height` of AVC standard
colourPrimaries	Same value as `colour_primaries` of AVC standard
transferCharacteristics	Same value as `transfer_characteristics` of AVC standard
matrixCoefficients	Same value as `matrix_coefficients` of AVC standard
videoFullRangeFlag	Same value as `video_full_range_flag` of AVC standard
reserved	Used within library (reserved area)
flag	Used within library (reserved area)
padding	Used within library (reserved area)
outputPts	Display time of the decode output picture

Description

This structure is used to store the optional picture information decoded and output when decoding with `sceAvcdecDecode()` in the AVC Decoder library.

For detailed information on each member in this structure, refer to the following AVC standard.

- ISO/IEC 14496-10:2014 Information technology -- Coding of audio-visual objects -- Part 10: Advanced Video Coding
  http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=66069

(The above reference destination has been confirmed as of November 12, 2014. Note that pages may have been subsequently moved or its contents modified.)

The display time PTS of the decode output picture is stored in *`outputPts`*. For details, refer to the "PTS Added to the Decode Output Result" section in the "AVC Decoder Overview" document.

See Also

`sceAvcdecDecode()`, `sceAvcdecDecodeStop()`

SceAvcdecInfoForInterlaced

Optional decoding information after AVC decoding (interlaced stream)

Definition

```
#include <videodec.h>
typedef struct SceAvcdecInfoForInterlaced{
        SceUInt32 numUnitsInTick;
        SceUInt32 timeScale;
        SceUChar8 fixedFrameRateFlag;

        SceUChar8 aspectRatioIdc;
        SceUShort16 sarWidth;
        SceUShort16 sarHeight;

        SceUChar8 colourPrimaries;
        SceUChar8 transferCharacteristics;
        SceUChar8 matrixCoefficients;

        SceUChar8 videoFullRangeFlag;

        SceUChar8 picStruct[2];
        SceUChar8 ctType;

        SceUChar8 flag;
        SceUChar8 padding[2];
        SceVideodecTimeStamp outputPts;

} SceAvcdecInfoForInterlaced;
```

Members

numUnitsInTick	Same value as `num_units_in_tick` of AVC standard
timeScale	Same value as `time_scale` of AVC standard
fixedFrameRateFlag	Same value as `fixed_frame_rate_flag` of AVC standard
aspectRatioIdc	Same value as `aspect_ratio_idc` of AVC standard
sarWidth	Same value as `sar_width` of AVC standard
sarHeight	Same value as `sar_height` of AVC standard
colourPrimaries	Same value as `colour_primaries` of AVC standard
transferCharacteristics	Same value as `transfer_characteristics` of AVC standard
matrixCoefficients	Same value as `matrix_coefficients` of AVC standard
videoFullRangeFlag	Same value as `video_full_range_flag` of AVC standard
picStruct	Same value as `pic_stuct` of AVC standard
ctType	Same value as `ct_type` of AVC standard, plus the `CT_TYPE_BIT` mask bit type
flag	Used within library (reserved area)
padding	Used within library (reserved area)
outputPts	The display time of the picture for decoding output, or the display time for the earlier field from among the two fields in an interlaced stream

Description

This structure is used to store the optional picture information decoded and output when decoding interlaced streams with `sceAvcdecDecode()` in the AVC Decoder library.

For detailed information on each member in this structure, refer to the following AVC standard.

- ISO/IEC 14496-10:2014 Information technology -- Coding of audio-visual objects -- Part 10: Advanced Video Coding
  http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=66069

(The above reference destination has been confirmed as of November 12, 2014. Note that pages may have been subsequently moved or its contents modified.)

In *`ctType`*, data required for the `CT_TYPE_BIT` mask bit type bit field with interlaced streams will be stored.

Bit Field	Description
`SCE_AVCDEC_CT_TYPE_BIT_ENABLE_PIC_STRUCT`	`1:pic_struct` exists `0:pic_struct` does not exist
`SCE_AVCDEC_CT_TYPE_BIT_ENABLE_CT_TYPE`	`1:ct_type` exists `0:ct_type` does not exist
`SCE_AVCDEC_CT_TYPE_BIT_FIELD_PAIR`	`1:field pair` `0:frame`
`SCE_AVCDEC_CT_TYPE_BIT_TOP_FIELD_FIRST`	`1:top field first` `0:bottom field first`

`SCE_AVCDEC_CT_TYPE_BIT_IDR_I_P_B_PIC_TYPE_FIELD`	`0:P-picture` or unknown
	`1:I-picture`
	`2:IDR-picture`
	`3:B-picture`

`SCE_AVCDEC_CT_TYPE_BIT_CT_TYPE_FIELD`	Same value as `ct_type` of AVC standard

In *`outputPts`*, the display time PTS for the picture for decoding output will be stored. For an interlaced stream, the display time PTS for the earlier field from among the two fields (top field and bottom field) will be stored.

See Also

`SceAvcdecInfo`, `SceAvcdecPictureForInterlaced`, CT_TYPE_BIT Mask Bit Type

SceAvcdecFrameOptionRGBA

Optional decoding RGBA information after AVC decoding

Definition

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

Members

alpha	Value of alpha plane when output as RGBA (If not specified, then 0xff.)
cscCoefficient	Specify the CSC coefficient type when converting from decoded output to RGBA
reserved	Used within library (reserved area)

Description

This structure is used to store the optional data of picture information decoded and output when decoding with `sceAvcdecDecode()` in the AVC Decoder library.

See Also

`sceAvcdecDecode()`, `sceAvcdecDecodeStop()`, Color Space Conversion Coefficient Type

SceAvcdecFrameOption

Optional decoding information after AVC decoding

Definition

```
#include <videodec.h>
typedef union SceAvcdecFrameOption{
        SceUInt8 reserved[16];
        SceAvcdecFrameOptionRGBA rgba;
} SceAvcdecFrameOption;
```

Members

reserved	Used within library (reserved area)
rgba	Specify when outputting from decoded output in RGBA with optional parameters

Description

This union is used to store the optional data of picture information decoded and output when decoding with `sceAvcdecDecode()` in the AVC Decoder library.

See Also

`sceAvcdecDecode(),sceAvcdecDecodeStop(),SceAvcdecFrameOptionRGBA`

SceAvcdecFrame

Decoding information after AVC decoding

Definition

```
#include <videodec.h>
typedef struct SceAvcdecFrame{
        SceUInt32 pixelType;
        SceUInt32 framePitch;
        SceUInt32 frameWidth;
        SceUInt32 frameHeight;

        SceUInt32 horizontalSize;
        SceUInt32 verticalSize;

        SceUInt32 frameCropLeftOffset;
        SceUInt32 frameCropRightOffset;
        SceUInt32 frameCropTopOffset;
        SceUInt32 frameCropBottomOffset;

        SceAvcdecFrameOption opt;

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

Members

pixelType	Specify pixel type constant
framePitch	Horizontal frame pitch of AVC decoding output destination (in pixels) Specify in multiples of 16, from 64 to 1920. (However, when *pixelType* is SCE_AVCDEC_PIXEL_YUV420_RASTER, *framePitch* must be in multiples of 32.) The *framePitch* x *frameHeight* area must be no greater than 1280 x 720.
frameWidth	Frame width of AVC decoding output destination (in pixels) Specify in multiples of 16, from 64 to 1920. The *framePitch* x *frameHeight* area must be no greater than 1280 x 720.
frameHeight	Frame height of AVC decoding output destination (in pixels) Specify in multiples of 16, from 64 to 1088. The *framePitch* x *frameHeight* area must be no greater than 1280 x 720.
horizontalSize	Horizontal size of AVC decoded result
verticalSize	Vertical size of AVC decoded result
frameCropLeftOffset	Returns offset value when FrameCrop occurs
frameCropRightOffset	Returns offset value when FrameCrop occurs
frameCropTopOffset	Returns offset value when FrameCrop occurs
frameCropBottomOffset	Returns offset value when FrameCrop occurs
opt	Optional decoding information after AVC decoding
pPicture[0]	Pointer to storage frame location for AVC decoding output
pPicture[1]	Pointer to later field when decoding an interlaced stream

Description

This structure is used to store the picture information decoded and output when decoding with `sceAvcdecDecode()` and `sceAvcdecDecodeStop()` in the AVC Decoder library.

To *`pixelType`*, specify the constant of the decoder output pixel type. When setting a value for *`opt`*, also specify `SCE_AVCDEC_OPTION_ENABLE`. For example, when specifying `SCE_AVCDEC_PIXEL_YUV420_RASTER` as well as `SCE_AVCDEC_OPTION_ENABLE`, specify them in the following way.

```
pixelType = SCE_AVCDEC_OPTION_ENABLE | SCE_AVCDEC_PIXEL_YUV420_RASTER;
```

To *`framePitch`*, set the horizontal frame pitch of the area storing the frame data of the decoded result, in pixels. Set this in multiples of 16, from 64 to 1920 (however, when *`pixelType`* is `SCE_AVCDEC_PIXEL_YUV420_RASTER`, in multiples of 32).

To *`frameWidth`*, set the frame width of the area storing the frame data of the decoded result, in pixels. Set this in multiples of 16, from 64 to 1920, with the value between one-fourth and four times the value of *`horizontalSize`*, the number of horizontal pixels of the AVC decoded result.

To *`frameHeight`*, set the frame height of the area storing the frame data of the decoded result, in pixels. Set this in multiples of 16, from 64 to 1088, with the value between one-fourth and four times the value of *`verticalSize`*, the number of vertical pixels of the AVC decoded result.

The *`framePitch`* x *`frameHeight`* area must be no greater than 1280 x 720.

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 in an uncached continuous physical address space (custom DRAM or physical continuous memory on the main memory).

When *`frameWidth`* or *`frameHeight`* of the storage destination frame of the AVC decoded result is smaller than *`horizontalSize`* or *`verticalSize`*, the number of horizontal or vertical pixels, of the AVC decoded result, the AVC decoded result is output according to *`frameWidth`* or *`frameHeight`*. When the AVC decoded result is the same value or less than *`frameWidth`* or *`frameHeight`*, the AVC decoded result is output with the number of pixels of *`horizontalSize`* or *`verticalSize`*.

Refer to "Frame Data of Decoded Result" in "AVC Decoder Overview" for details of the area for storing the frame data.

To *`pPicture`*`[1]`, assign NULL. When decoding a progressive stream, NULL will return as-is.

When decoding an interlaced stream, a pointer to the earlier field from among the two fields (top field and bottom field) will be output to *`pPicture`*`[0]`, and a pointer to the later field will be output to *`pPicture`*`[1]`.

The `SceAvcdecFrame` structure is a structure that receives the decode output from the library, but when performing `sceAvcdecDecode()` or `sceAvcdecDecodeStop()`, input the parameters in advance for *`pixelType`*, *`framePitch`*, *`frameWidth`*, *`frameHeight`*, and *`pPicture`*`[0]`, and if required input the parameter in advance for *`opt`*.

See Also

`sceAvcdecDecode()`, `sceAvcdecDecodeStop()`

SceAvcdecPicture

Decoding information after AVC decoding

Definition

```
#include <videodec.h>
typedef struct SceAvcdecPicture{
        SceSize size;
        SceAvcdecFrame frame;
        SceAvcdecInfo info;
} SceAvcdecPicture;
```

Members

size	Specify `sizeof(SceAvcdecPicture)`
frame	Stores picture information that is decoded and output
info	Stores optional decoding information that is decoded and output

Description

This structure is used to store the decoded output information when decoding with `sceAvcdecDecode()` in the AVC Decoder library.

The `SceAvcdecPicture` structure is a structure that receives the decode output from the library, but when performing `sceAvcdecDecode()` or `sceAvcdecDecodeStop()`, input the `sizeof(SceAvcdecPicture)` value for *size* in advance.

See Also

`sceAvcdecDecode()`, `sceAvcdecDecodeStop()`, `SceAvcdecFrame`, `SceAvcdecInfo`

SceAvcdecPictureForInterlaced

Decoding information after AVC decoding (interlaced stream)

Definition

```
#include <videodec.h>
typedef struct SceAvcdecPictureForInterlaced{
        SceSize size;
        SceAvcdecFrame frame;
        SceAvcdecInfoForInterlaced info;
} SceAvcdecPictureForInterlaced;
```

Members

size	Specify `sizeof(SceAvcdecPictureForInterlaced)`
frame	Stores picture information that is decoded and output
info	Stores optional decoding information that is decoded and output

Description

This structure is used to store the decoded output information when decoding interlaced streams with `sceAvcdecDecode()` in the AVC Decoder library.

The `SceAvcdecPictureForInterlaced` structure is a structure that receives the decode output of interlaced streams from the library. Input the `sizeof(SceAvcdecPictureForInterlaced)` value for *`size`* in advance.

To handle interlaced streams when executing `sceAvcdecDecode()` or `sceAvcdecDecodeStop()`, in *`pPicture`* in the `SceAvcdecArrayPicture` structure, input the start pointer value for a pointer array with a pointer to an `SceAvcdecPictureForInterlaced` structure stored instead of a pointer array with a pointer to an `SceAvcdecPicture` structure stored.

See Also

`sceAvcdecDecode()`, `sceAvcdecDecodeStop()`, `SceAvcdecFrame`, `SceAvcdecInfoForInterlaced`

SceAvcdecArrayPicture

Decoding information after AVC decoding

Definition

```
#include <videodec.h>
typedef struct SceAvcdecArrayPicture{
        SceUInt32 numOfOutput;
        SceUInt32 numOfElm;
        SceAvcdecPicture **pPicture;
} SceAvcdecArrayPicture;
```

Members

numOfOutput	Number of decoding information units decoded and output (0 or 1)
numOfElm	Number of storable picture information units decoded and output (specify 1)
pPicture	Pointer storing pointer array to decoding information that is decoded and output

Description

This structure is used to collectively handle the `SceAvcdecPicture` structure that stores decoded output information when decoding with `sceAvcdecDecode()` in the AVC Decoder library. This is used to receive the decoded result with `sceAvcdecDecode()` and `sceAvcdecDecodeStop()`.

The `SceAvcdecArrayPicture` structure is a structure that receives the decode output from the library, but when performing `sceAvcdecDecode()` or `sceAvcdecDecodeStop()`, input 1 for *numOfElm*, and input the beginning pointer value for the pointer array that stored the pointer to the `SceAvcdecPicture` structure for *pPicture*. For *pPicture*, when handling interlaced streams, input the start pointer value for the pointer array with a pointer to an `SceAvcdecPictureForInterlaced` structure stored.

See Also

`sceAvcdecDecode()`, `sceAvcdecDecodeStop()`, `SceAvcdecPicture`, `SceAvcdecFrame`, `SceAvcdecInfo`

Library Initialization / Termination

sceVideodecInitLibrary

Initialize video decoder

Definition

```
#include <videodec.h>
SceInt32 sceVideodecInitLibrary(
        SceUInt32 codecType,
        SceVideodecQueryInitInfo *pInitInfo
)
```

Argument

codecType	Constant indicating video decoder type (SCE_VIDEODEC_TYPE_HW_AVCDEC)
pInitInfo	Video decoder initialization parameter

Return Values

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

Description

This function initializes the video decoder.

To *codecType*, specify the constant indicating the video decoder type. When using an AVC decoder, specify the constant indicating the AVC decoder type (SCE_VIDEODEC_TYPE_HW_AVCDEC) and initialize the AVC decoder.

To *pInitInfo*, specify the initialization parameter structure of the video decoder. To use an AVC decoder, specify the initialization parameter structure of the AVC decoder of *hwAvc* in *pInitInfo*.

Follows the parameters specified with *pInitInfo* to create a video decode instance. At this time, the video decoder allocates a memory area of a maximum 6 MiB and with a 256-KiB alignment in an uncached continuous physical address space on the available main memory.

Example

```
SceVideodecQueryInitInfo queryInitMemInfo;

int res = sceVideodecInitLibrary
(SCE_VIDEODEC_TYPE_HW_AVCDEC,&queryInitMemInfo);
```

See Also

```
sceVideodecTermLibrary()
```

sceVideodecQueryMemSize

Return required size of the video decoder memory

Definition

```
#include <videodec.h>
SceInt32 sceVideodecQueryMemSize(
        SceUInt32 codecType,
        SceVideodecQueryInitInfo *pInitInfo,
        SceVideodecMemInfo *pMemInfo
)
```

Arguments

codecType	Constant indicating video decoder type (SCE_VIDEODEC_TYPE_HW_AVCDEC)
pInitInfo	Video decoder initialization parameter
pMemInfo	Video decoder memory information

Return Values

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

Description

This obtains the memory size required for initializing the video decoder.

To *codecType*, specify the constant indicating the video decoder type. When using an AVC decoder, specify the constant indicating the AVC decoder type (SCE_VIDEODEC_TYPE_HW_AVCDEC) and initialize the AVC decoder.

To *pInitInfo*, specify the initialization parameter structure of the video decoder. To use an AVC decoder, specify the initialization parameter structure of the AVC decoder of *hwAvc* in *pInitInfo*.

The memory size required for initializing the video decoder is stored to the *memSize* member of *pMemInfo*.

Examples

```
SceVideodecQueryInitInfo queryInitMemInfo;
SceVideodecMemInfo queryMemInfo;

int res = sceVideodecQueryMemSize
(SCE_VIDEODEC_TYPE_HW_AVCDEC,&queryInitMemInfo,&queryMemInfo);
```

See Also

SceVideodecQueryInitInfo, SceVideodecMemInfo,
sceVideodecInitLibraryWithUnmapMem()

sceVideodecInitLibraryWithUnmapMem

Initialize video decoder

Definition

```
#include <videodec.h>
SceInt32 sceVideodecInitLibraryWithUnmapMem(
        SceUInt32 codecType,
        SceVideodecCtrl *pCtrl,
        SceVideodecQueryInitInfo *pInitInfo
)
```

Arguments

codecType	Constant indicating video decoder type (SCE_VIDEODEC_TYPE_HW_AVCDEC)
pCtrl	Video decoder memory size information
pInitInfo	Video decoder initialization parameter

Return Values

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

Description

This function initializes the video decoder.

To *codecType*, specify the constant indicating the video decoder type. When using an AVC decoder, specify the constant indicating the AVC decoder type (SCE_VIDEODEC_TYPE_HW_AVCDEC) and initialize the AVC decoder.

Specify the amount of required memory obtained with sceVideodecQueryMemSize() or more to the *contextSize* member variable of *pCtrl*. Allocate the required memory size using sceCodecEngineAllocMemoryFromUnmapMemBlock() with 256 KiB alignment in an uncached continuous physical address space (custom DRAM or physical continuous memory on the main memory) and assign the starting address of the allocated memory to the *vaContext* member variable.

To *pInitInfo*, specify the same initialization parameter as that specified for the video decoder with sceVideodecQueryMemSize().

In the case of sceVideodecInitLibrary(), that function itself allocates the required memory in an uncached continuous physical address space on the main memory. On the other hand, in the case of sceVideodecInitLibraryWithUnmapMem(), the user must provide the required memory area.

Examples

```
SceVideodecCtrl videodecCtrl;
SceVideodecQueryInitInfo queryInitMemInfo;

int res = sceVideodecInitLibraryWithUnmapMem
(SCE_VIDEODEC_TYPE_HW_AVCDEC,&videodecCtrl,&queryInitMemInfo);
```

See Also

`SceVideodecCtrl`, `SceVideodecQueryInitInfo`, `sceVideodecQueryMemSize()`, `sceVideodecInitLibrary()`, `sceVideodecTermLibrary()`, `sceCodecEngineAllocMemoryFromUnmapMemBlock()`

sceVideodecTermLibrary

Terminate video decoder

Definition

```
#include <videodec.h>
SceInt32 sceVideodecTermLibrary(
        SceUInt32 codecType
)
```

Argument

codecType Constant indicating AVC decoder type (SCE_VIDEODEC_TYPE_HW_AVCDEC)

Return Values

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

Description

This function terminates the video decoder. To terminate an AVC decoder, specify SCE_VIDEODEC_TYPE_HW_AVCDEC to *codecType*.

Example

```
int res = sceVideodecTermLibrary(SCE_VIDEODEC_TYPE_HW_AVCDEC);
```

See Also

sceVideodecInitLibrary()

sceAvcdecSetInterlacedStreamMode

Set interlaced stream to be handled by AVC decoder

Definition

```
#include <videodec.h>
SceInt32 sceAvcdecSetInterlacedStreamMode(
        SceUInt32 codecType,
        SceUInt32 flag
)
```

Arguments

codecType	Constant indicating AVC decoder type (SCE_VIDEODEC_TYPE_HW_AVCDEC)
flag	Constant for enabling interlaced stream (SCE_AVCDEC_INTERLACED_STREAM_MODE_ENABLE)

Return Values

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

Description

Call this function when handling an interlaced stream with the AVC decoder. By calling it, both progressive and interlaced streams will be handled, but do not call it when using only progressive streams and not any interlaced streams.

Always call this function before executing sceAvcdecQueryDecoderMemSize().

To *codecType*, specify the constant indicating the AVC decoder type (SCE_VIDEODEC_TYPE_HW_AVCDEC)

For *flag*, specify the constant for enabling interlaced streams (SCE_AVCDEC_INTERLACED_STREAM_MODE_ENABLE).

When this function is called with 0 specified to *flag*, the default state (of being able to use progressive streams) will be set.

Examples

```
int res = sceAvcdecSetInterlacedStreamMode
(SCE_VIDEODEC_TYPE_HW_AVCDEC, SCE_AVCDEC_INTERLACED_STREAM_MODE_ENABLE);
```

See Also

SceAvcdecInfoForInterlaced, SceAvcdecPictureForInterlaced, sceAvcdecQueryDecoderMemSize(), sceAvcdecCreateDecoder(), Interlaced Stream Mode Type

sceAvcdecQueryDecoderMemSize

Return required size of the AVC decoder memory

Definition

```
#include <videodec.h>
SceInt32 sceAvcdecQueryDecoderMemSize(
        SceUInt32 codecType,
        const SceAvcdecQueryDecoderInfo *pDecoderInfo,
        SceAvcdecDecoderInfo *pMemInfo
)
```

Argument

codecType	Constant indicating AVC decoder type (SCE_VIDEODEC_TYPE_HW_AVCDEC)
pDecoderInfo	Initialization parameter of AVC decoder
pMemInfo	Returns required memory size of AVC decoder

Return Values

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

Description

This obtains the required memory size when initializing the AVC decoder.

To *codecType*, specify the constant indicating the AVC decoder type (SCE_VIDEODEC_TYPE_HW_AVCDEC).

To *pDecoderInfo*, specify the initialization parameter structure of the AVC decoder.

The memory size required for initializing the AVC decoder is stored to the *frameMemSize* member of *pMemInfo*.

Example

```
SceAvcdecQueryDecoderInfo queryDecoderInfo;
SceAvcdecDecoderInfo queryMemInfo;

int res = sceAvcdecQueryDecoderMemSize
(SCE_VIDEODEC_TYPE_HW_AVCDEC,&queryDecoderInfo, &queryMemInfo);
```

See Also

sceAvcdecCreateDecoder(),sceAvcdecDeleteDecoder()

sceAvcdecCreateDecoder

Create instance of AVC decoder

Definition

```
#include <videodec.h>
SceInt32 sceAvcdecCreateDecoder(
        SceUInt32 codecType,
        SceAvcdecCtrl *pCtrl,
        const SceAvcdecQueryDecoderInfo *pDecoderInfo
)
```

Argument

codecType	Constant indicating AVC decoder type (SCE_VIDEODEC_TYPE_HW_AVCDEC)
pCtrl	Instance of AVC decoder
pDecoderInfo	Initialization parameter of AVC decoder

Return Values

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

Description

This function creates an instance for initializing the AVC decoder and performing AVC decoding.

To *codecType*, specify the constant indicating the AVC decoder type (SCE_VIDEODEC_TYPE_HW_AVCDEC).

Specify the amount of required memory obtained with sceAvcdecQueryDecoderMemSize() or more to the *size* member variable of the *frameBuf* member variable of *pCtrl*, and assign the starting address of the memory allocated with 1-MiB alignment in an uncached continuous physical address space (custom DRAM or physical continuous memory on the main memory) to the *pBuf* member variable. However, specifying 0 to the *size* member variable and NULL to *pBuf* causes the AVC decoder to allocate the memory in the uncached continuous physical address space (custom DRAM) that is available in the memory space by the above conditions.

To *pDecoderInfo*, specify the same parameters specified with sceAvcdecQueryDecoderMemSize().

Example

```
SceAvcdecCtrl avcdecCtrl;
SceAvcdecQueryDecoderInfo queryDecoderInfo;

int res = sceAvcdecCreateDecoder
(SCE_VIDEODEC_TYPE_HW_AVCDEC,&avcdecCtrl, &queryDecoderInfo);
```

See Also

sceAvcdecQueryDecoderMemSize(),sceAvcdecDeleteDecoder()

sceAvcdecDeleteDecoder

Delete instance of AVC decoder

Definition

```
#include <videodec.h>
SceInt32 sceAvcdecDeleteDecoder(
        SceAvcdecCtrl *pCtrl
)
```

Argument

pCtrl Instance of AVC decoder

Return Values

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

Description

This function deletes the instance used to perform AVC decoding.

To *pCtrl*, specify the pointer to the `SceAvcdecCtrl` structure obtained with `sceAvcdecCreateDecoder()`.

Example

```
SceAvcdecCtrl avcdecCtrl;

int res = sceAvcdecDeleteDecoder (&avcdecCtrl);
```

See Also

`sceAvcdecQueryDecoderMemSize()`, `sceAvcdecCreateDecoder()`

AVC Decoding

sceAvcdecDecodeAvailableSize

Return available ES buffer space of AVC decoder

Definition

```
#include <videodec.h>
SceInt32 sceAvcdecDecodeAvailableSize(
        SceAvcdecCtrl *pCtrl
)
```

Argument

pCtrl Instance of AVC decoder

Return Values

Value	Description
>0	Available ES buffer space of AVC decoder
Negative number	Error (for details, see "Return Codes")

Description

This function returns the currently available ES buffer space of the AVC decoder.

To *pCtrl*, specify the AVC decode instance initialized with `sceAvcdecCreateDecoder()`.

When the return value is 0 or higher, the AVC decoder has available ES buffer space. When the ES data of the input AU with `sceAvcdecDecode()` is greater than the available ES buffer space, the `SCE_AVCDEC_ERROR_ES_BUFFER_FULL` error occurs.

Example

```
SceAvcdecCtrl avcdecCtrl;

unsigned int res = sceAvcdecDecodeAvailableSize (&avcdecCtrl);
```

See Also

`sceAvcdecDecode()`

sceAvcdecDecode

AVC decoding of one AU

Definition

```
#include <videodec.h>
SceInt32 sceAvcdecDecode(
        SceAvcdecCtrl *pCtrl,
        const SceAvcdecAu *pAu,
        SceAvcdecArrayPicture *pArrayPicture
)
```

Argument

pCtrl	Instance of AVC decoder
pAu	Input AU information of AVC decoder
pArrayPicture	Decoded output information after terminating AVC decoder

Return Values

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

Description

This function decodes one AU with the AVC decoder.

To *pCtrl*, specify the AVC decode instance initialized with `sceAvcdecCreateDecoder()`.

To *pAu*, specify the AVC ES data for one AU and PTS/DTS in units of 90 kHz. (Refer to "Time Stamp of Input Access Unit" in "AVC Decoder Overview" for details of PTS/DTS.)

When AVC decoding completes successfully, the AVC decoded result is stored to *pArrayPicture*. The value 1, which indicates the number of decoding information units, is entered to *numOfOutput* of the `SceAvcdecArrayPicture` structure, to which the *pArrayPicture* pointer points. In addition, the decoded output information is stored to the `SceAvcdecPicture` structure, which is the destination of the *pPicture* pointer of the `SceAvcdecArrayPicture` structure to which the *pArrayPicture* pointer points.

The decoded output might not be obtained even when decoding is performed with one AU specified to *pAu*. In this case, decoded output can be obtained by performing decoding several times. The decoded output is output with the display order guaranteed.

If the `SCE_AVCDEC_ERROR_ES_BUFFER_FULL` error occurs, the AU could not be input, so specify the same one AU and call `sceAvcdecDecode()` again.

To prevent the `SCE_AVCDEC_ERROR_ES_BUFFER_FULL` error from occurring, perform decoding while confirming that there is sufficient space by using `sceAvcdecDecodeAvailableSize()` before `sceAvcdecDecode()` to obtain the information of the available ES buffer space in the AVC decoder.

Unlike other errors, when the `SCE_AVCDEC_ERROR_ES_BUFFER_FULL` error occurs, the decoded output may be stored. Check the number of decoding information units in *numOfOutput* of the `SceAvcdecArrayPicture` structure.

To terminate stream decoding or to start playback from the start of a new group of pictures ("GOP"), call `sceAvcdecDecodeStop()` or `sceAvcdecDecodeFlush()`, and then call `sceAvcdecDeleteDecoder()` or `sceAvcdecDecode()`.

To obtain the decoded output result of the stream decoded by the AVC decoder, repeatedly call `sceAvcdecDecodeStop()`, not `sceAvcdecDecodeFlush()`, until *`numOfOutput`* of the `SceAvcdecArrayPicture` structure reaches 0. The decoded result remaining in the AVC decoder can be output. To delete the decoded result remaining in the AVC decoder, call `sceAvcdecDecodeFlush()`.

Refer to the `SceAvcdecAu` and `SceAvcdecFrame` structures and "Frame Data of Decoded Result" and "Buffer Restrictions for Positioning Data" in "AVC Decoder Overview" for details of the ES buffer storing one AU and the area for storing the frame data.

In addition, refer to the “Interlaced Streams” section in the ”AVC Decoder Overview” document when handling interlaced streams.

Example

```
SceAvcdecCtrl avcdecCtrl;
SceAvcdecAu decodeAu;
SceAvcdecArrayPicture arrayPicture;

int res = sceAvcdecDecode (&avcdecCtrl,  &decodeAu, &arrayPicture);
```

See Also

`sceAvcdecDecodeStop()`, `sceAvcdecDecodeFlush()`, `sceAvcdecDecodeAvailableSize()`

sceAvcdecDecodeStop

Stop AVC decoding

Definition

```
#include <videodec.h>
SceInt32 sceAvcdecDecodeStop(
        SceAvcdecCtrl *pCtrl,
        SceAvcdecArrayPicture *pArrayPicture
)
```

Argument

pCtrl	Instance of AVC decoder
pArrayPicture	Decoded output information after terminating AVC decoder

Return Values

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

Description

This function stops decoding of the AVC decoder.

To *pCtrl*, specify the AVC decode instance initialized with `sceAvcdecCreateDecoder()`.

The decoded result remaining in the AVC decoder is output to *pArrayPicture*. Repeatedly call this function until *numOfOutput* of the `SceAvcdecArrayPicture` structure reaches 0.

To terminate stream decoding or to start playback from the start of a new GOP, call `sceAvcdecDecodeStop()` or `sceAvcdecDecodeFlush()`, and then call `sceAvcdecDeleteDecoder()` or `sceAvcdecDecode()`.

To obtain the decoded output result of the stream decoded by the AVC decoder, repeatedly call `sceAvcdecDecodeStop()`, not `sceAvcdecDecodeFlush()`, until *numOfOutput* of the `SceAvcdecArrayPicture` structure reaches 0. The decoded result remaining in the AVC decoder can be output. To delete the decoded result remaining in the AVC decoder, call `sceAvcdecDecodeFlush()`.

Refer to the `SceAvcdecFrame` structure and "Frame Data of Decoded Result" and "Buffer Restrictions for Positioning Data" in "AVC Decoder Overview" document for details of the area for storing the frame data.

In addition, refer to the “Interlaced Streams” section in the ”AVC Decoder Overview” document when handling interlaced streams.

When handling `1field=1AU` type interlaced streams, an `SCE_AVCDEC_ERROR_INVALID_PICTURE` error will occur if `sceAvcdecDecodeStop()` is executed after decoding only 1 AU. Always input `2AU`. Use caution when implementing features that decode close to the GOP start, such as fast forward and rewind features.

Example

```
SceAvcdecCtrl avcdecCtrl;
SceAvcdecArrayPicture arrayPicture;

int res = sceAvcdecDecodeStop (&avcdecCtrl, &arrayPicture);
```

See Also

sceAvcdecDecode(),sceAvcdecDecodeFlush()

sceAvcdecDecodeFlush

Stop AVC decoding

Definition

```
#include <videodec.h>
SceInt32 sceAvcdecDecodeFlush(
        SceAvcdecCtrl *pCtrl
)
```

Argument

pCtrl Instance of AVC decoder

Return Values

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

Description

This function stops decoding of the AVC decoder.

To *pCtrl*, specify the AVC decode instance initialized with `sceAvcdecCreateDecoder()`.

To terminate stream decoding or to start playback from the start of a new GOP, call `sceAvcdecDecodeStop()` or `sceAvcdecDecodeFlush()`, and then call `sceAvcdecDeleteDecoder()` or `sceAvcdecDecode()`.

To obtain the decoded output result of the stream decoded by the AVC decoder, repeatedly call `sceAvcdecDecodeStop()`, not `sceAvcdecDecodeFlush()`, until *numOfOutput* of the `SceAvcdecArrayPicture` structure reaches 0. The decoded result remaining in the AVC decoder can be output. To delete the decoded result remaining in the AVC decoder, call `sceAvcdecDecodeFlush()`.

Example

```
SceAvcdecCtrl avcdecCtrl;

int res = sceAvcdecDecodeFlush (&avcdecCtrl);
```

See Also

`sceAvcdecDecodeStop()`, `sceAvcdecDecode()`

Constants

AVC Decoder Type

Constant indicating AVC decoder type

Definition

```
#define SCE_VIDEODEC_TYPE_HW_AVCDEC        (0x1001U)
```

Description

This constant indicates the AVC decoder type.

See Also

```
sceVideodecInitLibrary(),sceVideodecTermLibrary(),
sceAvcdecQueryDecoderMemSize(),sceAvcdecCreateDecoder()
```

Interlaced Stream Mode Type

Constant that enables interlaced stream

Definition

```
#define SCE_AVCDEC_INTERLACED_STREAM_MODE_ENABLE         (0x00000001U)
```

Description

This constant enables interlaced streams. Specify it for `sceAvcdecSetInterlacedStreamMode()`.

See Also

`sceAvcdecSetInterlacedStreamMode()`

Pixel Type

Constant indicating decoder output pixel type

Definition

```
#define SCE_AVCDEC_PIXEL_RGBA8888                (0x0U)
#define SCE_AVCDEC_PIXEL_BGRA8888                (0x8U)
#define SCE_AVCDEC_PIXEL_YUV420_RASTER           (0x10U)
#define SCE_AVCDEC_PIXEL_YUV420_PACKED_RASTER    (0x20U)
#define SCE_AVCDEC_OPTION_ENABLE                 (1<<31)
```

Description

This constant indicates the decoder output pixel type.

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

When specifying the *opt* member of the SceAvcdecFrame structure, also specify SCE_AVCDEC_OPTION_ENABLE.

For example, when specifying SCE_AVCDEC_PIXEL_YUV420_RASTER as well as SCE_AVCDEC_OPTION_ENABLE, specify them in the following way.

```
pixelType = SCE_AVCDEC_OPTION_ENABLE | SCE_AVCDEC_PIXEL_YUV420_RASTER;
```

See Also

SceAvcdecFrame, sceAvcdecDecode(), sceAvcdecDecodeStop()

Color Space Conversion Coefficient Type

Constant indicating coefficient type of color space conversion coefficient (CSC) of decoder output

Definition

```
#define SCE_AVCDEC_CSC_COEFFICIENT_DEFAULT        (0)
#define SCE_AVCDEC_CSC_COEFFICIENT_ITU601         (1)
#define SCE_AVCDEC_CSC_COEFFICIENT_ITU709         (2)
#define SCE_AVCDEC_CSC_COEFFICIENT_ITU601_FULL    (3)
#define SCE_AVCDEC_CSC_COEFFICIENT_ITU709_FULL    (4)
```

Description

This constant indicates the coefficient type when color space converting from YCbCr to RGBA during picture output at decoding and output.

Specify this to *cscCoefficient* of the SceAvcdecFrameOptionRGBA structure, which is the *rgba* member in the SceAvcdecFrameOption union, which is the *opt* member of the SceAvcdecFrame structure. When using this constant, also specify SCE_AVCDEC_OPTION_ENABLE to the *pixelType* member of the SceAvcdecFrame structure, and because the field of the SceAvcdecFrameOptionRGBA structure becomes valid, specify *cscCoefficient* together with other members.

See Also

SceAvcdecFrame, sceAvcdecDecode(), sceAvcdecDecodeStop()

CT_TYPE_BIT Mask Bit Type

Constants that indicate the mask bit for handling various data for *ctType* fields

Definition

```
#define SCE_AVCDEC_CT_TYPE_BIT_ENABLE_PIC_STRUCT         (0x80)
#define SCE_AVCDEC_CT_TYPE_BIT_ENABLE_CT_TYPE            (0x40)
#define SCE_AVCDEC_CT_TYPE_BIT_FIELD_PAIR                (0x20)
#define SCE_AVCDEC_CT_TYPE_BIT_TOP_FIELD_FIRST           (0x10)
#define SCE_AVCDEC_CT_TYPE_BIT_IDR_I_P_B_PIC_TYPE_FIELD  (0x0c)
#define SCE_AVCDEC_CT_TYPE_BIT_CT_TYPE_FIELD             (0x03)
```

Description

Data is stored bit by bit for *ctType* fields in the `SceAvcdecInfoForInterlaced` structure. These constants indicate the mask bit for handling the various data.

For details on the values that can be obtained with each mask, refer to the Description of the `SceAvcdecInfoForInterlaced` structure.

See Also

`SceAvcdecInfoForInterlaced`

Return Codes

List of return codes returned by the AVC Decoder library

Definition

Return codes returned by `sceVideodec*()` functions

Value	(Number)	Description
SCE_VIDEODEC_ERROR_INVALID_PARAM	0x80620802	Specified parameter is not appropriate
SCE_VIDEODEC_ERROR_OUT_OF_MEMORY	0x80620803	Not enough memory
SCE_VIDEODEC_ERROR_INVALID_STATE	0x80620804	A function was called from an inappropriate state
SCE_VIDEODEC_ERROR_ UNSUPPORT_IMAGE_SIZE	0x80620805	Image size is not supported
SCE_VIDEODEC_ERROR_ INVALID_COLOR_FORMAT	0x80620806	Color space format is not supported by color space conversion feature
SCE_VIDEODEC_ERROR_ NOT_PHY_CONTINUOUS_MEMORY	0x80620807	Physical address of specified memory area is not continuous
SCE_VIDEODEC_ERROR_ALREADY_USED	0x80620808	Already initialized
SCE_VIDEODEC_ERROR_INVALID_POINTER	0x80620809	Invalid pointer argument
SCE_VIDEODEC_ERROR_ES_BUFFER_FULL	0x8062080A	Unable to input ES to video decoder
SCE_VIDEODEC_ERROR_INITIALIZE	0x8062080B	Unable to initialize
SCE_VIDEODEC_ERROR_NOT_INITIALIZE	0x8062080C	Not initialized
SCE_VIDEODEC_ERROR_INVALID_STREAM	0x8062080D	Error in stream during decoding
SCE_VIDEODEC_ERROR_ INVALID_ARGUMENT_SIZE	0x8062080E	Value of `sizeof (structure)` is not specified in *`size`* member of structure

Return codes returned by `sceAvcdec*()` functions

Value	(Number)	Description
SCE_AVCDEC_ERROR_INVALID_PARAM	0x80620002	Specified parameter is not appropriate
SCE_AVCDEC_ERROR_OUT_OF_MEMORY	0x80620003	Not enough memory
SCE_AVCDEC_ERROR_INVALID_STATE	0x80620004	A function was called from an inappropriate state
SCE_AVCDEC_ERROR_ UNSUPPORT_IMAGE_SIZE	0x80620005	Image size is not supported
SCE_AVCDEC_ERROR_ INVALID_COLOR_FORMAT	0x80620006	Color space format is not supported by color space conversion feature
SCE_AVCDEC_ERROR_ NOT_PHY_CONTINUOUS_MEMORY	0x80620007	Physical address of specified memory area is not continuous
SCE_AVCDEC_ERROR_ALREADY_USED	0x80620008	Already initialized
SCE_AVCDEC_ERROR_INVALID_POINTER	0x80620009	Invalid pointer argument
SCE_AVCDEC_ERROR_ES_BUFFER_FULL	0x8062000A	Unable to input ES to AVC decoder
SCE_AVCDEC_ERROR_INITIALIZE	0x8062000B	Unable to initialize
SCE_AVCDEC_ERROR_NOT_INITIALIZE	0x8062000C	Not initialized
SCE_AVCDEC_ERROR_INVALID_STREAM	0x8062000D	Error in stream during decoding
SCE_AVCDEC_ERROR_ INVALID_ARGUMENT_SIZE	0x8062000E	Value of `sizeof (structure)` is not specified in *`size`* member of structure
SCE_AVCDEC_ERROR_ GREATER_THAN_1200_AT_LV30	0x8062000F	Input ES is Lv 3.0 or lower and the number of horizontal pixels exceed 1200
SCE_AVCDEC_ERROR_INTERLACED	0x80620010	Decoded stream was interlaced
SCE_AVCDEC_ERROR_INVALID_PICTURE	0x80620041	`sceAvcdecDecodeStop()` was called with an AU for only one of the fields input