libcamera Reference

PlayStation®Vita Programmer Tool Runtime Library Release 3.50

Table of Contents

Datatypes

SceCameraInfo

Stream setting information

Definition

```
typedef struct SceCameraInfo {
        SceSize sizeThis;
        SceUInt16 wPriority;
        SceUInt16 wFormat;
        SceUInt16 wResolution;
        SceUInt16 wFramerate;
        SceUInt16 wWidth;
        SceUInt16 wHeight;
        SceUInt16 wRange;
        SceUInt16 _padding_0;
        SceSize sizeIBase;
        SceSize sizeUBase;
        SceSize sizeVBase;
        void* pvIBase;
        void* pvUBase;
        void* pvVBase;
        SceUInt16 wPitch;
        SceUInt16 wBuffer;
} SceCameraInfo;
```

Members

sizeThis	Size of the `SceCameraInfo` structure
wPriority	Process priority
wFormat	Format
wResolution	Resolution
wFramerate	Frame rate
wWidth	Resolution width (`Read Only`)
wHeight	Resolution height (`Read Only`)
wRange	Data range
_padding_0	Unused
sizeIBase	Size of the IBase buffer
sizeUBase	Size of the UBase buffer
sizeVBase	Size of the VBase buffer
pvIBase	Address of the IBase buffer
pvUBase	Address of the UBase buffer
pvVBase	Address of the VBase buffer
wPitch	Byte count difference of user buffer and resolution width
wBuffer	User buffer setting method

Description

This structure contains the stream setting information required for opening the camera.
The application sets the each member when the camera is opened.

Set `sizeof(SceCameraInfo)`, the size of the `SceCameraInfo` structure, in *`sizeThis`*.
Always set `SCE_CAMERA_PRIORITY_SHARE` as the process priority in *`wPriority`*.

Process Priority Order	Description
`SCE_CAMERA_PRIORITY_SHARE`	The camera device is shared with other processes. Always choose this setting. If other processes using the camera become active, the present process will become inactive and calling `sceCameraStart()` and `sceCameraRead()` will succeed, but `SCE_CAMERA_STATUS_IS_NOT_ACTIVE` is set to *`dwStatus`* and images, etc., are not updated.
`SCE_CAMERA_PRIORITY_EXCLUSIVE`	This setting is invalid (an error).

Make one of the following settings as the format in *`wFormat`*.

Format	Description
`SCE_CAMERA_FORMAT_YUV422_PLANE`	Images are received from the camera in the YUV422 format. The data is divided into the YUV planes.
`SCE_CAMERA_FORMAT_YUV422_PACKED`	Images are received from the camera in the YUV422 format. The YUV is packed in pixel units in the data.
`SCE_CAMERA_FORMAT_YUV420_PLANE`	Images are received from the camera in the YUV422 format and converted to YUV 420 format inside the library. The data is divided into the YUV planes.
`SCE_CAMERA_FORMAT_YUV422_TO_ARGB`	Images are received from the camera in the YUV422 format and converted to the ARGB format inside the library.
`SCE_CAMERA_FORMAT_YUV422_TO_ABGR`	Images are received from the camera in the YUV422 format and converted to the ABGR format inside the library.
`SCE_CAMERA_FORMAT_RAW8`	Images are received in 8-bit RAWDATA format from the camera. However, it is only possible to select VGA (640x480) resolution. Also, it is not possible to set device attributes.

Set one of the following as the resolution in *`wResolution`*.

Resolution	Description
`SCE_CAMERA_RESOLUTION_VGA`	640x480
`SCE_CAMERA_RESOLUTION_QVGA`	320x240
`SCE_CAMERA_RESOLUTION_QQVGA`	160x120
`SCE_CAMERA_RESOLUTION_CIF`	352x288
`SCE_CAMERA_RESOLUTION_QCIF`	176x144
`SCE_CAMERA_RESOLUTION_PSP1`	480x272
`SCE_CAMERA_RESOLUTION_NGP1`	640x360

Set one of the following as frame rate in *wFramerate*.

Frame Rate	Description
SCE_CAMERA_FRAMERATE_3_75	3.75 fps
SCE_CAMERA_FRAMERATE_5	5 fps
SCE_CAMERA_FRAMERATE_7_5	7.5 fps
SCE_CAMERA_FRAMERATE_10	10 fps
SCE_CAMERA_FRAMERATE_15	15 fps
SCE_CAMERA_FRAMERATE_20	20 fps
SCE_CAMERA_FRAMERATE_24	24 fps
SCE_CAMERA_FRAMERATE_25	25 fps
SCE_CAMERA_FRAMERATE_30	30 fps
SCE_CAMERA_FRAMERATE_60	60 fps
SCE_CAMERA_FRAMERATE_120	120 fps The choice of resolution is limited to one of the following: SCE_CAMERA_RESOLUTION_QVGA (320x240) SCE_CAMERA_RESOLUTION_QQVGA (160x120) SCE_CAMERA_RESOLUTION_QCIF (176x144)

wWidth, *wHeight* are resolution width and height. At present, for reference purposes, libcamera sets the value.

Set one of the following as data range in *wRange*.

Frame Rate	Description
SCE_CAMERA_DATA_RANGE_FULL	Each color component is output in the full data range (0-255).
SCE_CAMERA_DATA_RANGE_BT601	Each color component is output by correcting the color according to the data range of the ITU-R BT.601 standard.

sizeIBase, *sizeUBase*, *sizeVBase*, *pvIBase*, *pvUBase*, *pvVBase* are information on the size and address of the user buffer. They are enabled when SCE_CAMERA_BUFFER_SETBYOPEN is selected as user buffer setting method in *wBuffer*.

The user buffer information set here is always used as a single buffer when calling sceCameraRead().

Specify the 16-byte aligned address of the continuous physical area in the user buffer. The memory secured from the CDRAM area is guaranteed to be a continuous physical area. To allocate a continuous physical area from the LPDDR area, specify a type for which physical continuity is guaranteed, such as SCE_KERNEL_MEMBLOCK_TYPE_USER_MAIN_PHYCONT_NC_RW when allocating memory. For details on the memory allocation, refer to "Memory Manager" in the "Kernel Reference" document. Moreover, the memory area with caches enabled can be used as a user buffer by specifying SCE_KERNEL_MEMBLOCK_TYPE_USER_MAIN_PHYCONT_RW, etc., but in this case, specify either SCE_CAMERA_FORMAT_YUV422_PLANE, SCE_CAMERA_FORMAT_YUV422_PACKED, or SCE_CAMERA_FORMAT_RAW8 as the format for use.

Set the following values in *sizeIBase* according to the format.

Format	***sizeIBase***
SCE_CAMERA_FORMAT_YUV422_PLANE SCE_CAMERA_FORMAT_YUV420_PLANE SCE_CAMERA_FORMAT_RAW8	Image width * height Example for VGA: 640*480=307200
SCE_CAMERA_FORMAT_YUV422_PACKED	Image width * height * 2 Example for VGA: 640*480*2=614400
SCE_CAMERA_FORMAT_YUV422_TO_ARGB SCE_CAMERA_FORMAT_YUV422_TO_ABGR	Image width * height * 4 Example for VGA: 640*480*4=1228800

Set the following values in *sizeUBase* according to the format.

Format	***sizeUBase***
SCE_CAMERA_FORMAT_YUV422_PLANE	Image (width/2) * height Example for VGA: 640*480/2=153600
SCE_CAMERA_FORMAT_YUV420_PLANE	Image (width/2) * (height/2) Example for VGA: 640*480/4=76800
SCE_CAMERA_FORMAT_YUV422_PACKED SCE_CAMERA_FORMAT_YUV422_TO_ARGB SCE_CAMERA_FORMAT_YUV422_TO_ABGR SCE_CAMERA_FORMAT_RAW8	0

Set the same value as *sizeUBase* in *sizeVBase*.

Set the address of the buffer allocated for each size in *pvIBase*, *pvUBase* and *pvVBase*. If the size is 0, set NULL.

When converting the format inside the library from YUV422 by selecting SCE_CAMERA_FORMAT_YUV422_TO_ARGB, SCE_CAMERA_FORMAT_YUV422_TO_ABGR or SCE_CAMERA_FORMAT_YUV420_PLANE for the format, set the byte count difference of the user buffer and resolution width in *wPitch*. When selecting a format other than those mentioned above and leaving the format inside the library unchanged, set 0.

Set one of the following values as user buffer setting method in *wBuffer*.

User Buffer Setting Method	Description
SCE_CAMERA_BUFFER_SETBYOPEN	Select to set the user buffer to the SceCameraInfo structure, which is passed to sceCameraOpen().
SCE_CAMERA_BUFFER_SETBYREAD	Select to set the user buffer to the SceCameraRead structure, which is passed to sceCameraRead().

See Also

sceCameraOpen()

SceCameraRead

Information of the read frame

Definition

```
typedef struct SceCameraRead {
        SceSize sizeThis;
        SceInt32 dwMode;
        SceInt32 dwExposureTimeMode;
        SceInt32 dwStatus;
        SceUInt64 qwFrame;
        SceUInt64 qwTimestamp;
        SceUInt32 dwExposureTime;
        SceUInt32 dwExposureTimeGap;
        SceUInt32 dwRaw8Format;
        SceUInt32 _padding_0;
        SceSize sizeIBase;
        SceSize sizeUBase;
        SceSize sizeVBase;
        void* pvIBase;
        void* pvUBase;
        void* pvVBase;
} SceCameraRead;
```

Members

sizeThis	Size of the `SceCameraRead` structure
dwMode	Reading mode setting
dwExposureTimeMode	Exposure time mode setting
dwStatus	Status when reading
qwFrame	Frame number
qwTimestamp	Timestamp (process time)
dwExposureTime	Exposure time (µs)
dwExposureTimeGap	Exposure time gap (µs)
dwRaw8Format	Memory allocation of RAW8 format (Bayer pattern)
_padding_0	Unused
sizeIBase	IBase buffer size
sizeUBase	UBase buffer size
sizeVBase	VBase buffer size
pvIBase	IBase buffer address
pvUBase	UBase buffer address
pvVBase	VBas buffer address

Description

This structure is used for receiving information on the frame when reading the most recent frame of the stream.

Set `sizeof(SceCameraRead)`, the size of the `SceCameraRead` structure, in *`sizeThis`*.

Set one of the following values as mode when reading in *dwMode*.

Read Mode Setting Method	Description
`SCE_CAMERA_READ_MODE_WAIT_NEXTFRAME_ON`	If the image data of the newest frame has been read, waits until the image from the camera is next updated and returns.
`SCE_CAMERA_READ_MODE_WAIT_NEXTFRAME_OFF`	If the image data of the newest frame has been read, returns immediately without waiting until the image from the camera is next updated.

For *dwExposureTimeMode*, set one of the following values as the exposure time mode during read access.

ExposureTime Obtainment Mode Setting Method	Description
`SCE_CAMERA_READ_GET_EXPOSURE_TIME_ON`	Obtains the exposure time when reading the image data of the latest frame is successful.
`SCE_CAMERA_READ_GET_EXPOSURE_TIME_OFF`	Does not obtain the exposure time when reading the image data of the latest frame is successful.

When calling `sceCameraRead()` is successful, *dwStatus*, *qwFrame*, *qwTimestamp* and *dwRaw8Format* are set and returned to the application. If `SCE_CAMERA_READ_GET_EXPOSURE_TIME_ON` has been set in *dwExposureTimeMode*, *dwExposureTime* and *dwExposureTimeGap* will also be set.

One of the following values is set to *dwStatus* at the time of normal termination.

Macro Constant	Description
`SCE_CAMERA_STATUS_IS_ACTIVE`	Indicates active status when reading
`SCE_CAMERA_STATUS_IS_NOT_ACTIVE`	Indicates inactive status when reading
`SCE_CAMERA_STATUS_IS_ALREADY_READ`	When reading, indicates that the frame has not been updated since the last reading.
`SCE_CAMERA_STATUS_IS_NOT_STABLE`	Indicates that images may not be stable when reading
`SCE_CAMERA_STATUS_IS_FORCED_STOP`	When reading, indicates that use of the camera has been disabled by the system
`SCE_CAMERA_STATUS_IS_FORCED_STOP_POWER_CONFIG_CHANGE`	When reading, indicates that use of the camera has been disabled by power configuration settings

One of the following values is set to *dwRaw8Format* at the time of normal termination.

Macro Constant	Description
`SCE_CAMERA_RAW8_FORMAT_UNKNOWN`	Indicates that a format other than `SCE_CAMERA_FORMAT_RAW8` is specified or that the format is an invalid RAW8 format.
`SCE_CAMERA_RAW8_FORMAT_BGGR`	Indicates RAW8 BGGR format
`SCE_CAMERA_RAW8_FORMAT_GRBG`	Indicates RAW8 GRBG format
`SCE_CAMERA_RAW8_FORMAT_RGGB`	Indicates RAW8 RGGB format
`SCE_CAMERA_RAW8_FORMAT_GBRG`	Indicates RAW8 GBRG format

For each format, refer to the "RAW8 Format" section of the "Reference Information" chapter in the "libcamera Overview" document.

sizeIBase, *sizeUBase*, *sizeVBase*, *pvIBase*, *pvUBase*, *pvVBase* are information on the size and address of the user buffer. They are enabled if SCE_CAMERA_BUFFER_SETBYREAD is set in *wBuffer* of the SceCameraInfo structure, which is passed to sceCameraOpen() as a user buffer setting method. If using multiple user buffers, switching from one to another every time sceCameraRead() is called, set them here.

Refer to the items of the SceCameraInfo structure for details.

For details on active/inactive status, refer to the sceCameraIsActive() section.

See Also

sceCameraRead()

Open/Close Functions

sceCameraOpen

Open camera

Definition

```
#include <camera.h>
int sceCameraOpen(
        int devnum,
        SceCameraInfo* pInfo
);
```

Arguments

devnum	Device number
pInfo	Stream setting information

Return Values

Returns SCE_OK for normal termination.

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

Macro Constant	Value	Description
SCE_CAMERA_ERROR_PARAM	0x802E0000	Parameter is invalid
SCE_CAMERA_ERROR_NOT_INIT	0x802E0002	Camera has not been initialized
SCE_CAMERA_ERROR_ALREADY_OPEN	0x802E0003	Camera is already open
SCE_CAMERA_ERROR_FORMAT_UNKNOWN	0x802E0007	Format is invalid
SCE_CAMERA_ERROR_RESOLUTION_UNKNOWN	0x802E0008	Resolution is invalid
SCE_CAMERA_ERROR_BAD_FRAMERATE	0x802E0009	Frame rate is invalid
SCE_CAMERA_ERROR_MAX_PROCESS	0x802E000D	Number of processes has reached maximum limit
SCE_CAMERA_ERROR_NOT_MOUNTED	0x802E0010	Camera is not mounted
SCE_CAMERA_ERROR_DATA_RANGE_UNKNOWN	0x802E0011	Data range is invalid

Description

This function opens the camera by specifying the device number of the camera and the stream setting information.

Set either SCE_CAMERA_DEVICE_FRONT or SCE_CAMERA_DEVICE_BACK as the device number in *devnum*.

For the method to set the SceCameraInfo structure indicated by *pInfo*, refer to the SceCameraInfo datatype.

See Also

SceCameraInfo

sceCameraClose

Close camera

Definition

```
#include <camera.h>
int sceCameraClose(
        int devnum
);
```

Arguments

devnum	Device number

Return Values

Returns `SCE_OK` for normal termination.

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

Macro Constant	Value	Description
`SCE_CAMERA_ERROR_PARAM`	0x802E0000	Parameter is invalid
`SCE_CAMERA_ERROR_NOT_INIT`	0x802E0002	Camera has not been initialized
`SCE_CAMERA_ERROR_NOT_OPEN`	0x802E0004	Camera is not open

Description

This function closes the camera.

Streaming Functions

sceCameraStart

Start stream

Definition

```
#include <camera.h>
int sceCameraStart(
        int devnum
);
```

Arguments

devnum Device number

Return Values

Returns SCE_OK for normal termination.

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

Macro Constant	Value	Description
SCE_CAMERA_ERROR_PARAM	0x802E0000	Parameter is invalid
SCE_CAMERA_ERROR_NOT_INIT	0x802E0002	Camera has not been initialized
SCE_CAMERA_ERROR_NOT_OPEN	0x802E0004	Camera is not open
SCE_CAMERA_ERROR_ALREADY_START	0x802E0005	Stream has already started
SCE_CAMERA_ERROR_FORMAT_UNKNOWN	0x802E0007	Format is invalid
SCE_CAMERA_ERROR_RESOLUTION_UNKNOWN	0x802E0008	Resolution is invalid
SCE_CAMERA_ERROR_BAD_FRAMERATE	0x802E0009	Frame rate is invalid
SCE_CAMERA_ERROR_DATA_RANGE_UNKNOWN	0x802E0011	Data range is invalid
SCE_CAMERA_ERROR_OTHER_ALREADY_START	0x802E0012	Another camera has already been in action

Description

This function starts the camera's stream.

Specify the device number of the opened camera in *devnum*.

sceCameraRead

Read stream

Definition

```
#include <camera.h>
int sceCameraRead(
        int devnum,
        SceCameraRead* pRead
);
```

Arguments

devnum	Device number
pRead	Pointer to structure with frame information

Return Values

Returns SCE_OK for normal termination.

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

Macro Constant	Value	Description
SCE_CAMERA_ERROR_PARAM	0x802E0000	Parameter is invalid
SCE_CAMERA_ERROR_NOT_INIT	0x802E0002	Camera has not been initialized
SCE_CAMERA_ERROR_NOT_OPEN	0x802E0004	Camera is not open
SCE_CAMERA_ERROR_NOT_START	0x802E0006	Stream has not started

Description

This function reads the image data of the most recent frame from the stream.

Set either SCE_CAMERA_DEVICE_FRONT or SCE_CAMERA_DEVICE_BACK as the device number in *devnum*.

For the method to set the SceCameraRead structure indicated by *pRead*, refer to the SceCameraRead datatype.

If SCE_CAMERA_READ_MODE_WAIT_NEXTFRAME_ON is set in *dwMode*, when reading image of the most recent frame, calling sceCameraRead() will be blocked until the image from the camera is next updated. Failure to update within a suitable interval for the frame rate that has been set will result in time-out. At this time, SCE_CAMERA_STATUS_IS_NOT_ACTIVE is set in *dwStatus*, but *qwFrame* and *qwTimestamp* are not set. If SCE_CAMERA_READ_MODE_WAIT_NEXTFRAME_OFF has been set, it will return immediately without waiting for the next update. At this time, SCE_CAMERA_STATUS_IS_ALREADY_READ is set in *dwStatus*. *qwFrame* and *qwTimestamp* are also set, but image data is not be copied to the user buffer.

If use of the camera has been disabled by the system, SCE_CAMERA_STATUS_IS_FORCED_STOP or SCE_CAMERA_STATUS_IS_FORCED_STOP_POWER_CONFIG_CHANGE will be set in *dwStatus*, while *qwFrame* and *qwTimestamp* will not be set. SCE_CAMERA_STATUS_IS_FORCED_STOP_POWER_CONFIG_CHANGE will be set if use of the camera has been disabled with the power configuration control. In other cases, SCE_CAMERA_STATUS_IS_FORCED_STOP will be set. In the current version of the SDK, it is set in cases such as when the system's screen (touchscreen) is in the non display state.

When image reading is terminated normally, `SCE_CAMERA_STATUS_IS_ACTIVE` is set in *`dwStatus`*. *`qwFrame`*, *`qwTimestamp`* and *`dwRaw8Format`* are also set. Moreover, immediately after the camera starts, etc., images may be obtainable even though the image output from the sensor may not yet be stable, but `SCE_CAMERA_STATUS_IS_NOT_STABLE` is returned. In this case, it is necessary to decide at the application level whether to display the obtained images. The time until the image output becomes stable depends on the brightness, etc., but it is about 10 frames in the worst case. If the camera is being used by system software or other applications, `SCE_CAMERA_STATUS_IS_NOT_ACTIVE` is set in *`dwStatus`*. However, *`qwFrame`*, *`qwTimestamp`* and *`dwRaw8Format`* are not set.

If `SCE_CAMERA_READ_GET_EXPOSURE_TIME_ON` was set in *`dwExposureTimeMode`*, *`dwExposureTime`* and *`dwExposureTimeGap`* are also set. Moreover, in the case of an environment in which the subject's brightness changes, the exposure time that can be obtained does not reflect the exposure time for the captured frame in real time, and a delay of approximately 1 second occurs until the exposure time the sensor actually used is reflected.

sceCameraStop

Stop stream

Definition

```
#include <camera.h>
int sceCameraStop(
        int devnum
);
```

Arguments

devnum Device number

Return Values

Returns `SCE_OK` for normal termination.

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

Macro Constant	Value	Description
`SCE_CAMERA_ERROR_PARAM`	0x802E0000	Parameter is invalid
`SCE_CAMERA_ERROR_NOT_INIT`	0x802E0002	Camera has not been initialized
`SCE_CAMERA_ERROR_NOT_OPEN`	0x802E0004	Camera is not open
`SCE_CAMERA_ERROR_NOT_START`	0x802E0006	Stream has not started

Description

This function stops the camera's stream.

sceCameraIsActive

Get active status

Definition

```
#include <camera.h>
int sceCameraIsActive(
        int devnum
);
```

Arguments

devnum Device number

Return Values

Returns 0 or 1 for normal termination.

1 indicates active status, while 0 indicates inactive status.

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

Macro Constant	Value	Description
SCE_CAMERA_ERROR_PARAM	0x802E0000	Parameter is invalid
SCE_CAMERA_ERROR_NOT_INIT	0x802E0002	Camera has not been initialized
SCE_CAMERA_ERROR_NOT_OPEN	0x802E0004	Camera is not open
SCE_CAMERA_ERROR_NOT_START	0x802E0006	Stream has not started

Description

This function obtains the camera's active status.

Active status indicates a state (return value: 1) whereby it is possible to obtain normally the frame image data from the camera. Inactive status indicates a state (return value: 0) whereby it is not possible to obtain frame image data from the camera used by own processes (such as cases in which system software or other applications are using the camera).

Functions to Obtain/Set Device Attributes

sceCameraGetSaturation

Get saturation

Definition

```
#include <camera.h>
int sceCameraGetSaturation(
        int devnum,
        int* pLevel
);
```

Arguments

devnum	Device number
pLevel	Pointer to the variable to receive saturation

Return Values

Returns `SCE_OK` for normal termination.

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

Macro Constant	Value	Description
`SCE_CAMERA_ERROR_PARAM`	0x802E0000	Parameter is invalid
`SCE_CAMERA_ERROR_NOT_OPEN`	0x802E0004	Camera is not open
`SCE_CAMERA_ERROR_NOT_START`	0x802E0006	Stream has not started

Description

It is a function to obtain the saturation set in the device.

Specify the device number of the opened camera in *devnum*.

In case of normal termination, one of the following values is set as the variable indicated by *pLevel*.

Macro Constant	Description
`SCE_CAMERA_ATTRIBUTE_SATURATION_0`	Saturation 0
`SCE_CAMERA_ATTRIBUTE_SATURATION_0_5`	Saturation 0.5
`SCE_CAMERA_ATTRIBUTE_SATURATION_1`	Saturation 1
`SCE_CAMERA_ATTRIBUTE_SATURATION_2`	Saturation 2
`SCE_CAMERA_ATTRIBUTE_SATURATION_3`	Saturation 3
`SCE_CAMERA_ATTRIBUTE_SATURATION_4`	Saturation 4

sceCameraSetSaturation

Set saturation

Definition

```
#include <camera.h>
int sceCameraSetSaturation(
        int devnum,
        int level
);
```

Arguments

devnum	Device number
level	Saturation

Return Values

Returns `SCE_OK` for normal termination.

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

Macro Constant	Value	Description
`SCE_CAMERA_ERROR_PARAM`	0x802E0000	Parameter is invalid
`SCE_CAMERA_ERROR_NOT_OPEN`	0x802E0004	Camera is not open
`SCE_CAMERA_ERROR_NOT_START`	0x802E0006	Stream has not started

Description

It is a function to set the saturation of the device.

Specify the device number of the opened camera in *devnum*.

Set one of the following values in *level*.

Macro Constant	Description
`SCE_CAMERA_ATTRIBUTE_SATURATION_0`	Saturation 0
`SCE_CAMERA_ATTRIBUTE_SATURATION_0_5`	Saturation 0.5
`SCE_CAMERA_ATTRIBUTE_SATURATION_1`	Saturation 1
`SCE_CAMERA_ATTRIBUTE_SATURATION_2`	Saturation 2
`SCE_CAMERA_ATTRIBUTE_SATURATION_3`	Saturation 3
`SCE_CAMERA_ATTRIBUTE_SATURATION_4`	Saturation 4

The initial saturation value is `SCE_CAMERA_ATTRIBUTE_SATURATION_1`.

sceCameraGetBrightness

Get Brightness

Definition

```
#include <camera.h>
int sceCameraGetBrightness(
        int devnum,
        int* pLevel
);
```

Arguments

devnum	Device number
pLevel	Pointer to the variable to receive brightness

Return Values

Returns `SCE_OK` for normal termination.

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

Macro Constant	Value	Description
`SCE_CAMERA_ERROR_PARAM`	0x802E0000	Parameter is invalid
`SCE_CAMERA_ERROR_NOT_OPEN`	0x802E0004	Camera is not open
`SCE_CAMERA_ERROR_NOT_START`	0x802E0006	Stream has not started

Description

It is a function to obtain the brightness set in the device.

Specify the device number of the opened camera in *devnum*.

For normal termination, a value of 0 to 255 is set to the variable pointed by *pLevel*.

sceCameraSetBrightness

Set brightness

Definition

```
#include <camera.h>
int sceCameraSetBrightness(
        int devnum,
        int level
);
```

Arguments

devnum	Device number
level	Brightness

Return Values

Returns `SCE_OK` for normal termination.

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

Macro Constant	Value	Description
`SCE_CAMERA_ERROR_PARAM`	0x802E0000	Parameter is invalid
`SCE_CAMERA_ERROR_NOT_OPEN`	0x802E0004	Camera is not open
`SCE_CAMERA_ERROR_NOT_START`	0x802E0006	Stream has not started

Description

It is a function to set the brightness of the device.
Specify the device number of the opened camera in *devnum*.

Set a value of 0 to 255 in *level*.
The initial brightness value is 128.

sceCameraGetContrast

Get contrast

Definition

```
#include <camera.h>
int sceCameraGetContrast(
        int devnum,
        int* pLevel
);
```

Arguments

devnum	Device number
pLevel	Pointer to the variable to receive contrast

Return Values

Returns SCE_OK for normal termination.

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

Macro Constant	Value	Description
SCE_CAMERA_ERROR_PARAM	0x802E0000	Parameter is invalid
SCE_CAMERA_ERROR_NOT_OPEN	0x802E0004	Camera is not open
SCE_CAMERA_ERROR_NOT_START	0x802E0006	Stream has not started

Description

It is a function to obtain the contrast set in the device.

Specify the device number of the opened camera in *devnum*.

For normal termination, a value of 0 to 255 is set to the variable pointed by *pLevel*.

sceCameraSetContrast

Set contrast

Definition

```
#include <camera.h>
int sceCameraSetContrast(
        int devnum,
        int level
);
```

Arguments

devnum	Device number
level	Contrast

Return Values

Returns SCE_OK for normal termination.

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

Macro Constant	Value	Description
SCE_CAMERA_ERROR_PARAM	0x802E0000	Parameter is invalid
SCE_CAMERA_ERROR_NOT_OPEN	0x802E0004	Camera is not open
SCE_CAMERA_ERROR_NOT_START	0x802E0006	Stream has not started

Description

It is a function to set the contrast of the device.

Specify the device number of the opened camera in *devnum*.

Set a value from 0 to 255 in *level*.

At present, the initial contrast value differs based on the device.

sceCameraGetSharpness

Get sharpness

Definition

```
#include <camera.h>
int sceCameraGetSharpness(
        int devnum,
        int* pLevel
);
```

Arguments

devnum	Device number
pLevel	Pointer to the variable to receive sharpness

Return Values

Returns `SCE_OK` for normal termination.

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

Macro Constant	Value	Description
SCE_CAMERA_ERROR_PARAM	0x802E0000	Parameter is invalid
SCE_CAMERA_ERROR_NOT_OPEN	0x802E0004	Camera is not open
SCE_CAMERA_ERROR_NOT_START	0x802E0006	Stream has not started

Description

It is a function to obtain the sharpness set in the device.

Specify the device number of the opened camera in *devnum*.

For normal termination, one of the following values is set to the variable pointed by *pLevel*.

Macro Constant	Description
SCE_CAMERA_ATTRIBUTE_SHARPNESS_1	Sharpness 100%
SCE_CAMERA_ATTRIBUTE_SHARPNESS_2	Sharpness 200%
SCE_CAMERA_ATTRIBUTE_SHARPNESS_3	Sharpness 300%
SCE_CAMERA_ATTRIBUTE_SHARPNESS_4	Sharpness 400%

sceCameraSetSharpness

Set sharpness

Definition

```
#include <camera.h>
int sceCameraSetSharpness(
        int devnum,
        int level
);
```

Arguments

devnum	Device number
level	Sharpness

Return Values

Returns SCE_OK for normal termination.

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

Macro Constant	Value	Description
SCE_CAMERA_ERROR_PARAM	0x802E0000	Parameter is invalid
SCE_CAMERA_ERROR_NOT_OPEN	0x802E0004	Camera is not open
SCE_CAMERA_ERROR_NOT_START	0x802E0006	Stream has not started

Description

It is a function to set the sharpness of the device.

Specify the device number of the opened camera in *devnum*.

Set one of the following values in *level*.

Macro Constant	Description
SCE_CAMERA_ATTRIBUTE_SHARPNESS_1	Sharpness 100%
SCE_CAMERA_ATTRIBUTE_SHARPNESS_2	Sharpness 200%
SCE_CAMERA_ATTRIBUTE_SHARPNESS_3	Sharpness 300%
SCE_CAMERA_ATTRIBUTE_SHARPNESS_4	Sharpness 400%

The initial sharpness value is SCE_CAMERA_ATTRIBUTE_SHARPNESS_1.

sceCameraGetReverse

Get mirror/flip

Definition

```
#include <camera.h>
int sceCameraGetReverse(
        int devnum,
        int* pMode
);
```

Arguments

devnum	Device number
pMode	Pointer to the variable to receive mirror/flip

Return Values

Returns `SCE_OK` for normal termination.

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

Macro Constant	Value	Description
`SCE_CAMERA_ERROR_PARAM`	0x802E0000	Parameter is invalid
`SCE_CAMERA_ERROR_NOT_OPEN`	0x802E0004	Camera is not open
`SCE_CAMERA_ERROR_NOT_START`	0x802E0006	Stream has not started

Description

This function obtains the mirror/flip set in the device.

Specify the device number of the opened camera in *devnum*.

For normal termination, one of the following values is set to the variable pointed by *pMode*.

Macro Constant	Description
`SCE_CAMERA_ATTRIBUTE_REVERSE_OFF`	Mirror OFF/Flip OFF
`SCE_CAMERA_ATTRIBUTE_REVERSE_MIRROR`	Mirror ON/Flip OFF
`SCE_CAMERA_ATTRIBUTE_REVERSE_FLIP`	Mirror OFF/Flip ON
`SCE_CAMERA_ATTRIBUTE_REVERSE_MIRROR_FLIP`	Mirror ON/Flip ON

sceCameraSetReverse

Set mirror/flip mode

Definition

```
#include <camera.h>
int sceCameraSetReverse(
        int devnum,
        int mode
);
```

Arguments

devnum	Device number
mode	Mirror/flip

Return Values

Returns SCE_OK for normal termination.

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

Macro Constant	Value	Description
SCE_CAMERA_ERROR_PARAM	0x802E0000	Parameter is invalid
SCE_CAMERA_ERROR_NOT_OPEN	0x802E0004	Camera is not open
SCE_CAMERA_ERROR_NOT_START	0x802E0006	Stream has not started

Description

This function sets the mirror/flip of the device.

Specify the device number of the opened camera in *devnum*.

Set one of the following values in *mode*.

Macro Constant	Description
SCE_CAMERA_ATTRIBUTE_REVERSE_OFF	Mirror OFF/Flip OFF
SCE_CAMERA_ATTRIBUTE_REVERSE_MIRROR	Mirror ON/Flip OFF
SCE_CAMERA_ATTRIBUTE_REVERSE_FLIP	Mirror OFF/Flip ON
SCE_CAMERA_ATTRIBUTE_REVERSE_MIRROR_FLIP	Mirror ON/Flip ON

The initial mirror/flip value is SCE_CAMERA_ATTRIBUTE_REVERSE_OFF.

sceCameraGetEffect

Get image effects

Definition

```
#include <camera.h>
int sceCameraGetEffect(
        int devnum,
        int* pMode
);
```

Arguments

devnum	Device number
pMode	Pointer to the variable to receive image effects

Return Values

Returns `SCE_OK` for normal termination.

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

Macro Constant	Value	Description
SCE_CAMERA_ERROR_PARAM	0x802E0000	Parameter is invalid
SCE_CAMERA_ERROR_NOT_OPEN	0x802E0004	Camera is not open
SCE_CAMERA_ERROR_NOT_START	0x802E0006	Stream has not started

Description

It is a function to obtain the image effects set in the device.

Specify the device number of the opened camera in *devnum*.

For normal termination, one of the following values is set to the variable pointed by *pMode*.

Macro Constant	Description
SCE_CAMERA_ATTRIBUTE_EFFECT_NORMAL	No image effects
SCE_CAMERA_ATTRIBUTE_EFFECT_NEGA	Negative image
SCE_CAMERA_ATTRIBUTE_EFFECT_BW	Black and white
SCE_CAMERA_ATTRIBUTE_EFFECT_SEPIA	Sepia tones
SCE_CAMERA_ATTRIBUTE_EFFECT_BLUISH	Bluish image
SCE_CAMERA_ATTRIBUTE_EFFECT_REDISH	Reddish image
SCE_CAMERA_ATTRIBUTE_EFFECT_GREENISH	Greenish image

sceCameraSetEffect

Set image effects

Definition

```
#include <camera.h>
int sceCameraSetEffect(
        int devnum,
        int mode
);
```

Arguments

devnum	Device number
mode	Image effect

Return Values

Returns SCE_OK for normal termination.

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

Macro Constant	Value	Description
SCE_CAMERA_ERROR_PARAM	0x802E0000	Parameter is invalid
SCE_CAMERA_ERROR_NOT_OPEN	0x802E0004	Camera is not open
SCE_CAMERA_ERROR_NOT_START	0x802E0006	Stream has not started

Description

It is a function to set the image effects of the device.

Specify the device number of the opened camera in *devnum*.

Set one of the following values in *mode*.

Macro Constant	Description
SCE_CAMERA_ATTRIBUTE_EFFECT_NORMAL	No image effects
SCE_CAMERA_ATTRIBUTE_EFFECT_NEGA	Negative image
SCE_CAMERA_ATTRIBUTE_EFFECT_BW	Black and white
SCE_CAMERA_ATTRIBUTE_EFFECT_SEPIA	Sepia tones
SCE_CAMERA_ATTRIBUTE_EFFECT_BLUISH	Bluish image
SCE_CAMERA_ATTRIBUTE_EFFECT_REDISH	Reddish image
SCE_CAMERA_ATTRIBUTE_EFFECT_GREENISH	Greenish image

The initial image effect value is SCE_CAMERA_ATTRIBUTE_EFFECT_NORMAL.

sceCameraGetEV

Get exposure compensation

Definition

```
#include <camera.h>
int sceCameraGetEV(
        int devnum,
        int* pLevel
);
```

Arguments

devnum	Device number
pLevel	Pointer to the variable to receive exposure compensation

Return Values

Returns SCE_OK for normal termination.

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

Macro Constant	Value	Description
SCE_CAMERA_ERROR_PARAM	0x802E0000	Parameter is invalid
SCE_CAMERA_ERROR_NOT_OPEN	0x802E0004	Camera is not open
SCE_CAMERA_ERROR_NOT_START	0x802E0006	Stream has not started

Description

It is a function to obtain the exposure compensation set in the device.

Specify the device number of the opened camera in *devnum*.

For normal termination, one of the following values is set as the variable indicated by *pLevel*.

Macro Constant	Description
SCE_CAMERA_ATTRIBUTE_EV_PLUS_2	EV +2.0
SCE_CAMERA_ATTRIBUTE_EV_PLUS_1_7	EV +1.7
SCE_CAMERA_ATTRIBUTE_EV_PLUS_1_5	EV +1.5
SCE_CAMERA_ATTRIBUTE_EV_PLUS_1_3	EV +1.3
SCE_CAMERA_ATTRIBUTE_EV_PLUS_1	EV +1.0
SCE_CAMERA_ATTRIBUTE_EV_PLUS_0_7	EV +0.7
SCE_CAMERA_ATTRIBUTE_EV_PLUS_0_5	EV +0.5
SCE_CAMERA_ATTRIBUTE_EV_PLUS_0_3	EV +0.3
SCE_CAMERA_ATTRIBUTE_EV_0	EV ±0
SCE_CAMERA_ATTRIBUTE_EV_MINUS_0_3	EV -0.3
SCE_CAMERA_ATTRIBUTE_EV_MINUS_0_5	EV -0.5
SCE_CAMERA_ATTRIBUTE_EV_MINUS_0_7	EV -0.7
SCE_CAMERA_ATTRIBUTE_EV_MINUS_1	EV -1.0
SCE_CAMERA_ATTRIBUTE_EV_MINUS_1_3	EV -1.3
SCE_CAMERA_ATTRIBUTE_EV_MINUS_1_5	EV -1.5
SCE_CAMERA_ATTRIBUTE_EV_MINUS_1_7	EV -1.7
SCE_CAMERA_ATTRIBUTE_EV_MINUS_2	EV -2.0

sceCameraSetEV

Set exposure compensation

Definition

```
#include <camera.h>
int sceCameraSetEV(
        int devnum,
        int level
);
```

Arguments

devnum	Device number
level	Exposure compensation

Return Values

Returns `SCE_OK` for normal termination.

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

Macro Constant	Value	Description
`SCE_CAMERA_ERROR_PARAM`	0x802E0000	Parameter is invalid
`SCE_CAMERA_ERROR_NOT_OPEN`	0x802E0004	Camera is not open
`SCE_CAMERA_ERROR_NOT_START`	0x802E0006	Stream has not started

Description

It is a function to set the exposure compensation of the device.

Specify the device number of the opened camera in *devnum*.

Set one of the following values in *level*.

Macro Constant	Description
`SCE_CAMERA_ATTRIBUTE_EV_PLUS_2`	EV +2.0
`SCE_CAMERA_ATTRIBUTE_EV_PLUS_1_7`	EV +1.7
`SCE_CAMERA_ATTRIBUTE_EV_PLUS_1_5`	EV +1.5
`SCE_CAMERA_ATTRIBUTE_EV_PLUS_1_3`	EV +1.3
`SCE_CAMERA_ATTRIBUTE_EV_PLUS_1`	EV +1.0
`SCE_CAMERA_ATTRIBUTE_EV_PLUS_0_7`	EV +0.7
`SCE_CAMERA_ATTRIBUTE_EV_PLUS_0_5`	EV +0.5
`SCE_CAMERA_ATTRIBUTE_EV_PLUS_0_3`	EV +0.3
`SCE_CAMERA_ATTRIBUTE_EV_0`	EV ±0
`SCE_CAMERA_ATTRIBUTE_EV_MINUS_0_3`	EV -0.3
`SCE_CAMERA_ATTRIBUTE_EV_MINUS_0_5`	EV -0.5
`SCE_CAMERA_ATTRIBUTE_EV_MINUS_0_7`	EV -0.7
`SCE_CAMERA_ATTRIBUTE_EV_MINUS_1`	EV -1.0
`SCE_CAMERA_ATTRIBUTE_EV_MINUS_1_3`	EV -1.3
`SCE_CAMERA_ATTRIBUTE_EV_MINUS_1_5`	EV -1.5
`SCE_CAMERA_ATTRIBUTE_EV_MINUS_1_7`	EV -1.7
`SCE_CAMERA_ATTRIBUTE_EV_MINUS_2`	EV -2.0

The initial exposure compensation value is `SCE_CAMERA_ATTRIBUTE_EV_0`.

sceCameraGetZoom

Get zoom level

Definition

```
#include <camera.h>
int sceCameraGetZoom(
        int devnum,
        int* pLevel
);
```

Arguments

devnum	Device number
pLevel	Pointer to the variable to receive zoom level

Return Values

Returns SCE_OK for normal termination.

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

Macro Constant	Value	Description
SCE_CAMERA_ERROR_PARAM	0x802E0000	Parameter is invalid
SCE_CAMERA_ERROR_NOT_OPEN	0x802E0004	Camera is not open
SCE_CAMERA_ERROR_NOT_START	0x802E0006	Stream has not started

Description

It is a function to obtain the zoom level set in the device.

Specify the device number of the opened camera in *devnum*.

For normal termination, a value of 10 to 40 is set to the variable pointed by *pLevel*.

This value is obtained by multiplying the actual zoom magnification of 1.0x to 4.0x by 10, converting it to an integer. For example, when this value is 15 the actual zoom magnification will be 1.5x.

With the zoom it is possible to enlarge the present resolution to VGA (640 x 480, 60 fps or less) size at the maximum. However, if the frame rate is 120 fps, the maximum limit will be restricted. Moreover, the maximum value of the zoom differs based on the present resolution.

For example, with QVGA (320 x 240) actual zoom magnification is up to 2.0x, while with QQVGA (160 x 120) it is up to 4.0x.

sceCameraSetZoom

Set the zoom level

Definition

```
#include <camera.h>
int sceCameraSetZoom(
        int devnum,
        int level
);
```

Arguments

devnum	Device number
level	Zoom level

Return Values

Returns SCE_OK for normal termination.

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

Macro Constant	Value	Description
SCE_CAMERA_ERROR_PARAM	0x802E0000	Parameter is invalid
SCE_CAMERA_ERROR_NOT_OPEN	0x802E0004	Camera is not open
SCE_CAMERA_ERROR_NOT_START	0x802E0006	Stream has not started

Description

It is a function to set the zoom level of the device.

Specify the device number of the opened camera in *devnum*.

Set a value of 10 to 40 in *level*. However, the maximum value differs based on the present resolution.

This value is obtained by multiplying the actual zoom magnification of 1.0x to 4.0x by 10, converting it to an integer.

For example, when actual zoom magnification is 1.5x this value will be 15.

With the zoom it is possible to enlarge the present resolution to the VGA (640 x 480, 60 fps or less) size at the maximum. However, if the frame rate is 120 fps, the maximum limit will be restricted.

Moreover, the maximum value of the zoom differs based on the present resolution and frame rate. The values which can be specified to *level* are as follows.

Present Resolution (60 fps or Less)	Value Specified in ***level***
SCE_CAMERA_RESOLUTION_VGA(640x480)	10
SCE_CAMERA_RESOLUTION_QVGA(320x240)	10 to 20
SCE_CAMERA_RESOLUTION_QQVGA(160x120)	10 to 40
SCE_CAMERA_RESOLUTION_CIF(352x288)	10 to 16
SCE_CAMERA_RESOLUTION_QCIF(176x144)	10 to 33
SCE_CAMERA_RESOLUTION_PSP1(480x272)	10 to 13
SCE_CAMERA_RESOLUTION_NGP1(640x360)	10

Present Resolution (120 fps)	Value Specified in ***level***
SCE_CAMERA_RESOLUTION_QVGA(320x240)	10
SCE_CAMERA_RESOLUTION_QQVGA(160x120)	10 to 20
SCE_CAMERA_RESOLUTION_QCIF(176x144)	10 to 16

The initial value of the zoom is 10.

sceCameraGetAntiFlicker

Get the anti-flicker

Definition

```
#include <camera.h>
int sceCameraGetAntiFlicker(
        int devnum,
        int* pMode
);
```

Arguments

devnum	Device number
pMode	Pointer to the variable to receive the anti-flicker

Return Values

Returns SCE_OK for normal termination.

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

Macro Constant	Value	Description
SCE_CAMERA_ERROR_PARAM	0x802E0000	Parameter is invalid
SCE_CAMERA_ERROR_NOT_OPEN	0x802E0004	Camera is not open
SCE_CAMERA_ERROR_NOT_START	0x802E0006	Stream has not started

Description

This function obtains the anti-flicker set in the device.

Specify the device number of the opened camera in *devnum*.

For normal termination, one of the following values is set to the variable pointed by *pMode*.

Macro Constant	Description
SCE_CAMERA_ATTRIBUTE_ANTIFLICKER_AUTO	Automatic
SCE_CAMERA_ATTRIBUTE_ANTIFLICKER_50HZ	50 Hz
SCE_CAMERA_ATTRIBUTE_ANTIFLICKER_60HZ	60 Hz

sceCameraSetAntiFlicker

Set anti-flicker

Definition

```
#include <camera.h>
int sceCameraSetAntiFlicker(
        int devnum,
        int mode
);
```

Arguments

devnum	Device number
mode	Anti-flicker

Return Values

Returns `SCE_OK` for normal termination.

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

Macro Constant	Value	Description
SCE_CAMERA_ERROR_PARAM	0x802E0000	Parameter is invalid
SCE_CAMERA_ERROR_NOT_OPEN	0x802E0004	Camera is not open
SCE_CAMERA_ERROR_NOT_START	0x802E0006	Stream has not started

Description

This function sets the anti-flicker of the device.

Specify the device number of the opened camera in *devnum*.

Set one of the following values in *mode*.

Macro Constant	Description
SCE_CAMERA_ATTRIBUTE_ANTIFLICKER_AUTO	Automatic
SCE_CAMERA_ATTRIBUTE_ANTIFLICKER_50HZ	50 Hz
SCE_CAMERA_ATTRIBUTE_ANTIFLICKER_60HZ	60 Hz

The initial anti-flicker value is `SCE_CAMERA_ATTRIBUTE_ANTIFLICKER_AUTO`.

sceCameraGetISO

Get ISO speed

Definition

```
#include <camera.h>
int sceCameraGetISO(
        int devnum,
        int* pMode
);
```

Arguments

devnum	Device number
pMode	Pointer to the variable to receive ISO speed

Return Values

Returns `SCE_OK` for normal termination.

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

Macro Constant	Value	Description
`SCE_CAMERA_ERROR_PARAM`	0x802E0000	Parameter is invalid
`SCE_CAMERA_ERROR_NOT_OPEN`	0x802E0004	Camera is not open
`SCE_CAMERA_ERROR_NOT_START`	0x802E0006	Stream has not started

Description

This function obtains the ISO speed set in the device.

Specify the device number of the opened camera in *devnum*.

For normal termination, one of the following values is set to the variable pointed by *pMode*.

Macro Constant	Description
`SCE_CAMERA_ATTRIBUTE_ISO_AUTO`	Automatic
`SCE_CAMERA_ATTRIBUTE_ISO_100`	ISO100/21°
`SCE_CAMERA_ATTRIBUTE_ISO_200`	ISO200/24°
`SCE_CAMERA_ATTRIBUTE_ISO_400`	ISO400/27°

sceCameraSetISO

Set ISO speed

Definition

```
#include <camera.h>
int sceCameraSetISO(
        int devnum,
        int mode
);
```

Arguments

devnum	Device number
mode	ISO speed

Return Values

Returns SCE_OK for normal termination.

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

Macro Constant	Value	Description
SCE_CAMERA_ERROR_PARAM	0x802E0000	Parameter is invalid
SCE_CAMERA_ERROR_NOT_OPEN	0x802E0004	Camera is not open
SCE_CAMERA_ERROR_NOT_START	0x802E0006	Stream has not started

Description

This function sets the ISO speed of the device.

Specify the device number of the opened camera in *devnum*.

Set one of the following values in *mode*.

Macro Constant	Description
SCE_CAMERA_ATTRIBUTE_ISO_AUTO	Automatic
SCE_CAMERA_ATTRIBUTE_ISO_100	ISO100/21°
SCE_CAMERA_ATTRIBUTE_ISO_200	ISO200/24°
SCE_CAMERA_ATTRIBUTE_ISO_400	ISO400/27°

The initial ISO speed value is SCE_CAMERA_ATTRIBUTE_ISO_AUTO.

sceCameraGetWhiteBalance

Get white balance

Definition

```
#include <camera.h>
int sceCameraGetWhiteBalance(
        int devnum,
        int* pMode
);
```

Arguments

devnum	Device number
pMode	Pointer to the variable to receive white balance

Return Values

Returns SCE_OK for normal termination.

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

Macro Constant	Value	Description
SCE_CAMERA_ERROR_PARAM	0x802E0000	Parameter is invalid
SCE_CAMERA_ERROR_NOT_OPEN	0x802E0004	Camera is not open
SCE_CAMERA_ERROR_NOT_START	0x802E0006	Stream has not started

Description

It is a function to obtain the white balance set in the device.

Specify the device number of the opened camera in *devnum*.

For normal termination, one of the following values is set to the variable pointed by *pMode*.

Macro Constant	Description
SCE_CAMERA_ATTRIBUTE_WHITEBALANCE_AUTO	Automatic
SCE_CAMERA_ATTRIBUTE_WHITEBALANCE_DAY	Daylight
SCE_CAMERA_ATTRIBUTE_WHITEBALANCE_CWF	Cool White Fluorescent
SCE_CAMERA_ATTRIBUTE_WHITEBALANCE_A	Standard light source A (Tungsten)

sceCameraSetWhiteBalance

Set white balance

Definition

```
#include <camera.h>
int sceCameraSetWhiteBalance(
        int devnum,
        int mode
);
```

Arguments

devnum	Device number
mode	White balance

Return Values

Returns SCE_OK for normal termination.

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

Macro Constant	Value	Description
SCE_CAMERA_ERROR_PARAM	0x802E0000	Parameter is invalid
SCE_CAMERA_ERROR_NOT_OPEN	0x802E0004	Camera is not open
SCE_CAMERA_ERROR_NOT_START	0x802E0006	Stream has not started

Description

It is a function to set the white balance of the device.

Specify the device number of the opened camera in *devnum*.

Set one of the following values in *mode*.

Macro Constant	Description
SCE_CAMERA_ATTRIBUTE_WHITEBALANCE_AUTO	Automatic
SCE_CAMERA_ATTRIBUTE_WHITEBALANCE_DAY	Daylight
SCE_CAMERA_ATTRIBUTE_WHITEBALANCE_CWF	Cool White Fluorescent
SCE_CAMERA_ATTRIBUTE_WHITEBALANCE_A	Standard light source A (Tungsten)

The initial white balance value is SCE_CAMERA_ATTRIBUTE_WHITEBALANCE_AUTO.

sceCameraGetBacklight

Get backlight compensation

Definition

```
#include <camera.h>
int sceCameraGetBacklight(
        int devnum,
        int* pMode
);
```

Arguments

devnum	Device number
pMode	Pointer to the variable to receive backlight compensation

Return Values

Returns `SCE_OK` for normal termination.

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

Macro Constant	Value	Description
`SCE_CAMERA_ERROR_PARAM`	0x802E0000	Parameter is invalid
`SCE_CAMERA_ERROR_NOT_OPEN`	0x802E0004	Camera is not open
`SCE_CAMERA_ERROR_NOT_START`	0x802E0006	Stream has not started

Description

It is a function to obtain the backlight compensation set in the device.

Specify the device number of the opened camera in *devnum*.

For normal termination, one of the following values is set as the variable indicated by *pMode*.

Macro Constant	Description
`SCE_CAMERA_ATTRIBUTE_BACKLIGHT_OFF`	Backlight compensation OFF
`SCE_CAMERA_ATTRIBUTE_BACKLIGHT_ON`	Backlight compensation ON

sceCameraSetBacklight

Set backlight compensation

Definition

```
#include <camera.h>
int sceCameraSetBacklight(
        int devnum,
        int mode
);
```

Arguments

devnum	Device number
mode	Backlight compensation

Return Values

Returns `SCE_OK` for normal termination.

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

Macro Constant	Value	Description
`SCE_CAMERA_ERROR_PARAM`	0x802E0000	Parameter is invalid
`SCE_CAMERA_ERROR_NOT_OPEN`	0x802E0004	Camera is not open
`SCE_CAMERA_ERROR_NOT_START`	0x802E0006	Stream has not started

Description

It is a function to set the backlight compensation of the device.

Specify the device number of the opened camera in *devnum*.

Set one of the following values in *mode*.

Macro Constant	Description
`SCE_CAMERA_ATTRIBUTE_BACKLIGHT_OFF`	Backlight compensation OFF
`SCE_CAMERA_ATTRIBUTE_BACKLIGHT_ON`	Backlight compensation ON

The initial backlight compensation value is `SCE_CAMERA_ATTRIBUTE_BACKLIGHT_OFF`.

sceCameraGetNightmode

Get night mode

Definition

```
#include <camera.h>
int sceCameraGetNightmode(
        int devnum,
        int* pMode
);
```

Arguments

devnum	Device number
pMode	Pointer to the variable to receive night mode

Return Values

Returns SCE_OK for normal termination.

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

Macro Constant	Value	Description
SCE_CAMERA_ERROR_PARAM	0x802E0000	Parameter is invalid
SCE_CAMERA_ERROR_NOT_OPEN	0x802E0004	Camera is not open
SCE_CAMERA_ERROR_NOT_START	0x802E0006	Stream has not started

Description

It is a function to obtain the night mode set in the device.

Specify the device number of the opened camera in *devnum*.

For normal termination, set one of the following values to the variable pointed by *pMode*.

Macro Constant	Description
SCE_CAMERA_ATTRIBUTE_NIGHTMODE_OFF	Night mode OFF
SCE_CAMERA_ATTRIBUTE_NIGHTMODE_LESS10	10 lux or below
SCE_CAMERA_ATTRIBUTE_NIGHTMODE_LESS100	100 lux or below
SCE_CAMERA_ATTRIBUTE_NIGHTMODE_OVER100	100 lux or over

sceCameraSetNightmode

Set night mode

Definition

```
#include <camera.h>
int sceCameraSetNightmode(
        int devnum,
        int mode
);
```

Arguments

devnum	Device number
mode	Night mode

Return Values

Returns SCE_OK for normal termination.

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

Macro Constant	Value	Description
SCE_CAMERA_ERROR_PARAM	0x802E0000	Parameter is invalid
SCE_CAMERA_ERROR_NOT_OPEN	0x802E0004	Camera is not open
SCE_CAMERA_ERROR_NOT_START	0x802E0006	Stream has not started

Description

It is a function to set the night mode of the device.

Specify the device number of the opened camera in *devnum*.

Set one of the following values in *mode*.

Macro Constant	Description
SCE_CAMERA_ATTRIBUTE_NIGHTMODE_OFF	Night mode OFF
SCE_CAMERA_ATTRIBUTE_NIGHTMODE_LESS10	10 lux or below
SCE_CAMERA_ATTRIBUTE_NIGHTMODE_LESS100	100 lux or below
SCE_CAMERA_ATTRIBUTE_NIGHTMODE_OVER100	100 lux or over

The initial night mode value is SCE_CAMERA_ATTRIBUTE_NIGHTMODE_OFF.

In night mode, brightness is adjusted by changing the frame rate in accordance with the brightness of the photography environment. For this reason, frame rate will decrease as follows with each setting value:

- Maximum of around 1/4 in the case of SCE_CAMERA_ATTRIBUTE_NIGHTMODE_LESS10
- Maximum of around 1/3 in the case of SCE_CAMERA_ATTRIBUTE_NIGHTMODE_LESS100
- Maximum of around 1/2 in the case of SCE_CAMERA_ATTRIBUTE_NIGHTMODE_OVER100

For example, if SCE_CAMERA_ATTRIBUTE_NIGHTMODE_LESS10 is specified when frame rate is 60 fps, frame rate will vary in the 60 fps - approx. 15 fps range.

sceCameraGetAutoControlHold

Get auto control hold status

Definition

```
#include <camera.h>
int sceCameraGetAutoControlHold(
        int devnum,
        int* pMode
);
```

Arguments

devnum	Device number
pMode	Pointer to the variable to receive the auto control hold status

Return Values

Returns `SCE_OK` for normal termination.

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

Macro Constant	Value	Description
`SCE_CAMERA_ERROR_PARAM`	0x802E0000	Parameter is invalid
`SCE_CAMERA_ERROR_NOT_OPEN`	0x802E0004	Camera is not open
`SCE_CAMERA_ERROR_NOT_START`	0x802E0006	Stream has not started

Description

It is a function to obtain the auto control hold status set to the device.

Specify the device number of the opened camera in *devnum*.

For normal termination, set one of the following values to the variable pointed by *pMode*.

Macro Constant	Description
`SCE_CAMERA_ATTRIBUTE_AUTOCONTROLHOLD_OFF`	Auto control hold OFF
`SCE_CAMERA_ATTRIBUTE_AUTOCONTROLHOLD_ON`	Auto control hold ON

sceCameraSetAutoControlHold

Set auto control hold status

Definition

```
#include <camera.h>
int sceCameraSetAutoControlHold(
        int devnum,
        int mode
);
```

Arguments

devnum	Device number
mode	Auto control hold mode

Return Values

Returns SCE_OK for normal termination.

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

Macro Constant	Value	Description
SCE_CAMERA_ERROR_PARAM	0x802E0000	Parameter is invalid
SCE_CAMERA_ERROR_NOT_OPEN	0x802E0004	Camera is not open
SCE_CAMERA_ERROR_NOT_START	0x802E0006	Stream has not started

Description

It is a function to set the auto control hold status of the device.

Specify the device number of the opened camera in *devnum*.

Set one of the following values in *mode*.

Macro Constant	Description
SCE_CAMERA_ATTRIBUTE_AUTOCONTROLHOLD_OFF	Auto control hold OFF
SCE_CAMERA_ATTRIBUTE_AUTOCONTROLHOLD_ON	Auto control hold ON

The default value of the auto control hold status is SCE_CAMERA_ATTRIBUTE_AUTOCONTROLHOLD_OFF.

If SCE_CAMERA_ATTRIBUTE_AUTOCONTROLHOLD_ON is specified for this function, auto control of Auto Exposure Control (AEC), Auto Gain Control (AGC), and Auto White Balance (AWB) is stopped and hold is implemented with AEC, AGC, and AWB at this time. If SCE_CAMERA_ATTRIBUTE_AUTOCONTROLHOLD_OFF is specified, hold is canceled and auto control is executed.

Owing to the specifications of the sensor, if a function such as sceCameraSetISO() or sceCameraSetWhiteBalance() that conflicts with the feature that is held by this function is used at the same time as this function, the function that was set last is enabled. For example, if SCE_CAMERA_ATTRIBUTE_WHITEBALANCE_AUTO was set with sceCameraSetWhiteBalance() after hold was executed by setting SCE_CAMERA_ATTRIBUTE_AUTOCONTROLHOLD_ON, the hold status is maintained for AEC and AGC, but auto control is done for AWB. Moreover, if following hold, SCE_CAMERA_ATTRIBUTE_ISO_AUTO is set with sceCameraSetISO(), the hold status is maintained for AWB but auto control is done for AEC and AGC.

sceCameraGetExposureCeiling

Get exposure ceiling

Definition

```
#include <camera.h>
int sceCameraGetExposureCeiling(
        int devnum,
        int* pMode
);
```

Arguments

devnum	Device number
pMode	Pointer to the variable to receive exposure ceiling

Return Values

Returns `SCE_OK` for normal termination.

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

Macro Constant	Value	Description
`SCE_CAMERA_ERROR_PARAM`	0x802E0000	Parameter is invalid
`SCE_CAMERA_ERROR_NOT_OPEN`	0x802E0004	Camera is not open
`SCE_CAMERA_ERROR_NOT_START`	0x802E0006	Stream has not started

Description

It is a function to obtain the exposure ceiling set in the device.

Specify the device number of the opened camera in *devnum*.

For normal termination, set one of the following values to the variable pointed by *pMode*.

Macro Constant	Description
`SCE_CAMERA_ATTRIBUTE_EXPOSURE_CEILING_NORMAL`	Normal state
`SCE_CAMERA_ATTRIBUTE_EXPOSURE_CEILING_1_2`	Half of the normal state

sceCameraSetExposureCeiling

Set exposure ceiling

Definition

```
#include <camera.h>
int sceCameraSetExposureCeiling(
        int devnum,
        int mode
);
```

Arguments

devnum	Device number
mode	Exposure ceiling

Return Values

Returns `SCE_OK` for normal termination.

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

Macro Constant	Value	Description
SCE_CAMERA_ERROR_PARAM	0x802E0000	Parameter is invalid
SCE_CAMERA_ERROR_NOT_OPEN	0x802E0004	Camera is not open
SCE_CAMERA_ERROR_NOT_START	0x802E0006	Stream has not started

Description

It is a function to set the exposure ceiling of the device.

Specify the device number of the opened camera in *devnum*.

Set one of the following values in *mode*.

Macro Constant	Description
SCE_CAMERA_ATTRIBUTE_EXPOSURE_CEILING_NORMAL	Normal state
SCE_CAMERA_ATTRIBUTE_EXPOSURE_CEILING_1_2	Half of the normal state

The initial exposure ceiling value is `SCE_CAMERA_ATTRIBUTE_EXPOSURE_CEILING_NORMAL`.

Exposure ceiling setting is valid only when the frame rate is set to `SCE_CAMERA_FRAMERATE_60`. An error will occur if the setting is attempted when another frame rate is specified.

When the exposure ceiling is set to `SCE_CAMERA_ATTRIBUTE_EXPOSURE_CEILING_1_2`, the camera device behaves so that the maximum time of exposure compensated by the device will be half of the normal state. In addition, since the exposure ceiling has dependence on the anti flicker setting, which can be set with `sceCameraSetAntiFlicker()`, setting the exposure ceiling to `SCE_CAMERA_ATTRIBUTE_EXPOSURE_CEILING_1_2` has an effect of reducing image quality variations caused by the differences of the anti flicker setting (50 or 60Hz).

For example, if the frame rate is set to `SCE_CAMERA_FRAMERATE_60` and the anti flicker setting is `SCE_CAMERA_ATTRIBUTE_ANTIFLICKER_60HZ`, the maximum time of exposure in the normal state is 16.6ms, twice the cardinal number of 60Hz (8.3ms). If `SCE_CAMERA_ATTRIBUTE_EXPOSURE_CEILING_1_2` is set during this state, the camera device behaves so that the maximum time of exposure will be 8.3ms, half of the normal state. On the other hand, when the anti flicker setting is `SCE_CAMERA_ATTRIBUTE_ANTIFLICKER_50HZ`, the maximum time of exposure in the normal state is around 10ms, the same value as the cardinal number of 50Hz (10ms). Therefore, the maximum time of exposure will remain around 10ms even if `SCE_CAMERA_ATTRIBUTE_EXPOSURE_CEILING_1_2` is set. When the anti flicker setting is `SCE_CAMERA_ATTRIBUTE_ANTIFLICKER_AUTO`, the camera device behaves in accordance with the frequency detected by the device.

Functions to Obtain/Set Information

sceCameraGetDeviceLocation

Get physical location information of device

Definition

```
#include <camera.h>
int sceCameraGetDeviceLocation(
        int devnum,
        SceFVector3* pLocation
);
```

Arguments

devnum	Device number
pLocation	Pointer of variable for receiving physical location

Return Values

Returns SCE_OK for normal termination.

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

Macro Constant	Value	Description
SCE_CAMERA_ERROR_PARAM	0x802E0000	Parameter is invalid
SCE_CAMERA_ERROR_NOT_INIT	0x802E0002	Camera has not been initialized

Description

This function obtains the physical location information of where the device is.

Specify the device number of the camera in *devnum*.

Upon normal termination, the distance (mm) from the origin (center point of screen surface) is set with the X, Y and Z axes for the variable indicated by *pLocation*. The accuracy of the obtained value is ±1 mm, and the maximum/minimum values are ±300 mm.

Constants

Return Codes

Return codes returned by functions of libcamera

Definition

Macro Constant	Value	Description
SCE_CAMERA_ERROR_PARAM	0x802E0000	Parameter is invalid
SCE_CAMERA_ERROR_NOT_INIT	0x802E0002	Camera is not initialized
SCE_CAMERA_ERROR_ALREADY_OPEN	0x802E0003	Camera is already open
SCE_CAMERA_ERROR_NOT_OPEN	0x802E0004	Camera is not open
SCE_CAMERA_ERROR_ALREADY_START	0x802E0005	Stream has already started
SCE_CAMERA_ERROR_NOT_START	0x802E0006	Stream has not started
SCE_CAMERA_ERROR_FORMAT_UNKNOWN	0x802E0007	Format is invalid
SCE_CAMERA_ERROR_RESOLUTION_UNKNOWN	0x802E0008	Resolution is invalid
SCE_CAMERA_ERROR_BAD_FRAMERATE	0x802E0009	Frame rate is invalid
SCE_CAMERA_ERROR_TIMEOUT	0x802E000A	Time has run out
SCE_CAMERA_ERROR_ATTRIBUTE_UNKNOWN	0x802E000C	Device attribute is invalid
SCE_CAMERA_ERROR_MAX_PROCESS	0x802E000D	Number of processes has reached maximum limit.
SCE_CAMERA_ERROR_NOT_MOUNTED	0x802E0010	Camera is not mounted
SCE_CAMERA_ERROR_DATA_RANGE_UNKNOWN	0x802E0011	Data range is invalid
SCE_CAMERA_ERROR_OTHER_ALREADY_START	0x802E0012	Another camera has already been in action
SCE_CAMERA_ERROR_FATAL	0x802E00FF	Other fatal error occurred