VRmUsbCam C API

Version 3.3.0.0
3/18/2014 3:05:00 PM

Table of Contents

Module Index

Modules

Here is a list of all modules:

File Index

File List

Here is a list of all files with brief descriptions:

Module Documentation

Basic Types

Typedefs

- typedef unsigned char **VRmBYTE**
- typedef unsigned short int **VRmWORD**
- typedef unsigned int **VRmDWORD**
- typedef unsigned int **VRmBOOL**
- typedef const char * **VRmSTRING**
- typedef **VRM_STRUCT _VRmSizeI**
- typedef **VRM_STRUCT _VRmPointI**
- typedef **VRM_STRUCT _VRmRectI**

Variables

- int **m_height**
- **VRmSizeI**
- int **m_y**
- **VRmPointI**
- int **m_top**
- int **m_width**
- **VRmRectI**

Detailed Description

Typedef Documentation

typedef VRM_STRUCT _VRmPointI

```
Initial value:{
        int m_x
```

"point" type (int)

typedef VRM_STRUCT _VRmRectI

```
Initial value:{
        int m_left
```

"rect" type (int)

typedef VRM_STRUCT _VRmSizeI

```
Initial value:{
        int m_width
```

"size" type (int)

typedef unsigned int VRmBOOL

typedef unsigned char VRmBYTE

typedef unsigned int VRmDWORD

typedef const char* VRmSTRING

typedef unsigned short int VRmWORD

Variable Documentation

VRmDWORD m_height

height in lines

int m_top

int m_width

int m_y

VRmPointI

VRmRectI

VRmSizeI

Error Handling / General Management

Typedefs

- typedef **VRM_ENUM _VRmRetVal**
- typedef **VRM_ENUM _VRmErrorCode**

Functions

- **VRM_EXTERN VRmSTRING VRM_API VRmUsbCamGetLastError** (void)
- **VRM_EXTERN** int **VRM_API VRmUsbCamGetLastErrorCode** (void)
- **VRM_EXTERN** void **VRM_API VRmUsbCamClearLastError** (void)
- **VRM_EXTERN VRmBOOL VRM_API VRmUsbCamLastErrorWasTriggerTimeout** (void)
- **VRM_EXTERN VRmBOOL VRM_API VRmUsbCamLastErrorWasTriggerStall** (void)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamEnableLogging** (void)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamEnableLoggingEx** (**VRmSTRING** f_log_file_name)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetVersion** (**VRmDWORD** *fp_version)
- **VRM_EXTERN** void **VRM_API VRmUsbCamCleanup** (void)

Detailed Description

Typedef Documentation

typedef VRM_ENUM _VRmErrorCode

```
Initial value:{

        VRM_ERROR_CODE_SUCCESS                                  =       0x00000000,

        VRM_ERROR_CODE_FUNCTION_CALL_TIMEOUT    =       0x00040003,


        VRM_ERROR_CODE_GENERIC_ERROR                    =       0x80004005,

        VRM_ERROR_CODE_TRIGGER_TIMEOUT                  =       0x80040001,
        VRM_ERROR_CODE_TRIGGER_STALL                    =       0x80040002,
        VRM_ERROR_CODE_TRANSFER_TIMEOUT                 =       0x80040004

} VRmErrorCode
```

error codes used by get last error code

typedef VRM_ENUM _VRmRetVal

```
Initial value:{

        VRM_FAILED=0,

        VRM_SUCCESS=1
} VRmRetVal
```

return value values of API function calls

Function Documentation

VRM_EXTERN void VRM_API VRmUsbCamCleanup (void)

you should call this at application exit in order to cleanup all resources left over from VRmUsbCam API.

AFTER THAT, NO OTHER API FUNCTIONS MUST BE CALLED ANY LONGER!

easiest way is to use the following call at application init:

```
atexit(VRmUsbCamCleanup);
```

VRM_EXTERN void VRM_API VRmUsbCamClearLastError (void)

reset the error code and error string to success.

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamEnableLogging (void)

enable logging.

for customer support, enable the logging facilities of the VRmUsbCam library. you may want to add following lines to your source code:

```
...
```

```
#ifdef _DEBUG
   VRmUsbCamEnableLogging();
#endif
...
```

NOTE: Only the first successfull call to VRmUsbCamEnableLogging or VRmUsbCamEnableLoggingEx will have an effect. Subsequent calls will be ignored.

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamEnableLoggingEx (VRmSTRING *f_log_file_name*)

enable logging with selectable file name for customer support, enable the logging facilities of the VRmUsbCam library.

The file "f_log_file_name" will be newly created. you may want to add following lines to your source code:

```
...
#ifdef _DEBUG
   VRmUsbCamEnableLoggingEx( LOG_FILE_NAME );
#endif
...
```

NOTE: Only the first successful call to VRmUsbCamEnableLogging or VRmUsbCamEnableLoggingEx will have an effect. Subsequent calls will be ignored.

VRM_EXTERN VRmSTRING VRM_API VRmUsbCamGetLastError (void)

retrieve the error string of the last function call.

if an API function fails (return value is VRM_FAILED) use this function to retrieve an error description as C string. NOTE: the returned string is only guaranteed to be valid until the next API call

VRM_EXTERN int VRM_API VRmUsbCamGetLastErrorCode (void)

retrieve the error code of the last function call.

if an API function fails (return is value VRM_FAILED) use this function to retrieve an error error code as int. NOTE: the returned code is only guaranteed to be valid until the next API call

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetVersion (VRmDWORD * *fp_version*)

get the version of the API.

the version number is represented as decimal integer with 4 digits, ie. API version v2.3.0.0 is represented as decimal 2300.

VRM_EXTERN VRmBOOL VRM_API VRmUsbCamLastErrorWasTriggerStall (void)

last error was trigger stall.

if an API function fails (return value VRM_FAILED) use this function to check if error was a trigger stall

VRM_EXTERN VRmBOOL VRM_API VRmUsbCamLastErrorWasTriggerTimeout (void)

last error was trigger timeout.

if an API function fails (return value VRM_FAILED) use this function to check if error was a trigger timeout

Device Management

Macros

- #define **VRM_VRMUSBCAMDEVICE_DEFINED**

Typedefs

- typedef struct
- VRmUsbCamDeviceInternal * **VRmUsbCamDevice**
- typedef **VRM_STRUCT _VRmDeviceKey**

Functions

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamUpdateDeviceKeyList** (void)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamUpdateDeviceKeyListEx** (**VRmBOOL** f_local, **VRmBOOL** f_usb, **VRmBOOL** f_ethernet)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetDeviceKeyListSize** (**VRmDWORD** *fp_size)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetDeviceKeyListEntry** (**VRmDWORD** f_index, **VRmDeviceKey** **fpp_device_key)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetVendorId** (const **VRmDeviceKey** *fcp_device_key, **VRmWORD** *fp_vendor_id)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetProductId** (const **VRmDeviceKey** *fcp_device_key, **VRmWORD** *fp_product_id)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetGroupId** (const **VRmDeviceKey** *fcp_device_key, **VRmWORD** *fp_group_id)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetSerialString** (const **VRmDeviceKey** *fcp_device_key, **VRmSTRING** *fp_serial_str)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetIpAddress** (const **VRmDeviceKey** *fcp_device_key, **VRmSTRING** *fp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetLocalIpAddress** (const **VRmDeviceKey** *fcp_device_key, **VRmSTRING** *fp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamCompareDeviceKeys** (const **VRmDeviceKey** *fcp_device_key1, const **VRmDeviceKey** *fcp_device_key2, **VRmBOOL** *fp_result)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamFreeDeviceKey** (**VRmDeviceKey** **fpp_device_key)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamOpenDevice** (const **VRmDeviceKey** *fcp_device_key, **VRmUsbCamDevice** *fp_device)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetDeviceKey** (**VRmUsbCamDevice** f_device, **VRmDeviceKey** **fpp_device_key)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamCloseDevice** (**VRmUsbCamDevice** f_device)

Variables

- **VRmSTRING mp_manufacturer_str**
- **VRmSTRING mp_product_str**
- **VRmBOOL m_busy**
- void * **mp_private**
- **VRmDeviceKey**

Detailed Description

Macro Definition Documentation

#define VRM_VRMUSBCAMDEVICE_DEFINED

device handle.

this handle represents a device, create using **VRmUsbCamOpenDevice()**, release using **VRmUsbCamCloseDevice()**

Typedef Documentation

typedef VRM_STRUCT _VRmDeviceKey

```
Initial value:{

        VRmDWORD m_serial
```

struct to identify devices.

device key is a unique combination of serial, manufacturer and product string NOTE: the strings within this struct are only guaranteed to be valid until the next call to **VRmUsbCamUpdateDeviceKeyList()**!

typedef struct VRmUsbCamDeviceInternal* VRmUsbCamDevice

Function Documentation

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamCloseDevice (VRmUsbCamDevice *f_device*)

close device using device handle

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamCompareDeviceKeys (const VRmDeviceKey * *fcp_device_key1*, const VRmDeviceKey * *fcp_device_key2*, VRmBOOL * *fp_result*)

compare two device keys.

sets target of fp_result to 0 if keys are equal, sets it to 1 otherwise

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamFreeDeviceKey (VRmDeviceKey ** *fpp_device_key*)

free device key received by **VRmUsbCamGetDeviceKeyListEntry()**

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetDeviceKey (VRmUsbCamDevice *f_device*, VRmDeviceKey ** *fpp_device_key*)

get device key for device in use.

use **VRmUsbCamFreeDeviceKey()** to free the key when you're done

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetDeviceKeyListEntry (VRmDWORD *f_index*, VRmDeviceKey ** *fpp_device_key*)

get device key by index.

returns the device key by index = [0...number of attached devices-1], use **VRmUsbCamFreeDeviceKey()** to free the key when you're done

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetDeviceKeyListSize (VRmDWORD * *fp_size*)

get number of attached devices

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetGroupId (const VRmDeviceKey * *fcp_device_key*, VRmWORD * *fp_group_id*)

get group id (16bit) of device key

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetIpAddress (const VRmDeviceKey * *fcp_device_key*, VRmSTRING * *fp_value*)

get IP address string of device key (for non-ethernet devices, this is an empty string).

note: the returned string is only guaranteed to be valid until the next API call!

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetLocalIpAddress (const VRmDeviceKey * *fcp_device_key*, VRmSTRING * *fp_value*)

get IP address string of local interface that is to be used for communication with device of device key (for non-ethernet devices, this is an empty string).

note: the returned string is only guaranteed to be valid until the next API call!

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetProductId (const VRmDeviceKey * *fcp_device_key*, VRmWORD * *fp_product_id*)

get product id (16bit) of device key

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetSerialString (const VRmDeviceKey * *fcp_device_key*, VRmSTRING * *fp_serial_str*)

get serial string of device key.

note: the returned string is only guaranteed to be valid until the next call to **VRmUsbCamUpdateDeviceKeyList()**!

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetVendorId (const VRmDeviceKey * *fcp_device_key*, VRmWORD * *fp_vendor_id*)

get vendor id (16bit) of device key

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamOpenDevice (const VRmDeviceKey * *fcp_device_key*, VRmUsbCamDevice * *fp_device*)

open device.

open device using device key returned by **VRmUsbCamGetDeviceKeyListEntry()**, to release fp_device use **VRmUsbCamCloseDevice()**

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamUpdateDeviceKeyList (void)

search for compatible devices.

if your application wants to support PnP, you should call this function periodically, at least once every 5 seconds, and handle the PnP events using static callback handlers, see **VRmUsbCamRegisterStaticCallback()**

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamUpdateDeviceKeyListEx (VRmBOOL *f_local*, VRmBOOL *f_usb*, VRmBOOL *f_ethernet*)

search for compatible devices.

use parameters to control which devices are listed. f_local: list local devices (on intelligent camera) f_usb: list devices connected via USB f_ethernet: list devices connected via Ethernet if your application wants to support PnP, you should call this function periodically, at least once every 5 seconds, and handle the PnP events using static callback handlers, see **VRmUsbCamRegisterStaticCallback()**

Variable Documentation

VRmBOOL m_busy

busy means the device is already used by another application

VRmSTRING mp_manufacturer_str

manufacturer name string

void * mp_private

private = additional internal data

VRmSTRING mp_product_str

product name string

VRmDeviceKey

User Data Storage / Non-Volatile Memory

Typedefs

- typedef **VRM_STRUCT _VRmUserData**

Functions

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamLoadUserData** (**VRmUsbCamDevice** f_device, **VRmUserData** **fpp_user_data)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSaveUserData** (**VRmUsbCamDevice** f_device, const **VRmUserData** *fcp_user_data)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamNewUserData** (**VRmUserData** **fpp_user_data, **VRmDWORD** f_length)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamFreeUserData** (**VRmUserData** **fpp_user_data)

Variables

- **VRmBYTE** * **mp_data**
- **VRmUserData**

Detailed Description

Typedef Documentation

typedef VRM_STRUCT _VRmUserData

```
Initial value:{

        VRmDWORD m_length
```

struct for user data storage

Function Documentation

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamFreeUserData (VRmUserData ** *fpp_user_data*)

free VRmUserData created by **VRmUsbCamNewUserData()** and **VRmUsbCamLoadUserData()**

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamLoadUserData (VRmUsbCamDevice *f_device*, VRmUserData ** *fpp_user_data*)

load user data from eeprom.

use **VRmUsbCamFreeUserData()** to delete afterwards. if no data was saved before this function returns without error but returns a VRmUserData with length = 0

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamNewUserData (VRmUserData ** *fpp_user_data*, VRmDWORD *f_length*)

allocate new user data.

length in bytes, use **VRmUsbCamFreeUserData()** to delete

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamSaveUserData (VRmUsbCamDevice *f_device*, const VRmUserData * *fcp_user_data*)

save user data in eeprom.

NOTE: you can use the VRM_PROPID_DEVICE_NV_MEM_FREE_I property to determine the number of bytes free in the non-volatile memory of the device

---

Variable Documentation

VRmBYTE* mp_data

pointer to user data

VRmUserData

Timer

Functions

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamRestartTimer** (void)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetCurrentTime** (double *fp_current_time)

Detailed Description

Function Documentation

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetCurrentTime (double * *fp_current_time*)

get current timestamp from timer

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamRestartTimer (void)

restart timer for timestamps, use for all open devices

Image Handling

Macros

- #define **VRM_IMAGE_BUFFER_SIZE_MAX** (0xFFFFFFFF)

Typedefs

- typedef **VRM_ENUM _VRmColorFormat**
- typedef **VRM_ENUM _VRmImageModifier**
- typedef **VRM_STRUCT _VRmImageFormat**
- typedef **VRM_STRUCT _VRmImage**

Functions

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamNewImage** (**VRmImage** **fpp_image, **VRmImageFormat** f_image_format)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamCopyImage** (**VRmImage** **fpp_image, const **VRmImage** *fcp_src_image)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamCropImage** (**VRmImage** **fpp_image, const **VRmImage** *fcp_src_image, const **VRmRectI** *fcp_roi)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSetImage** (**VRmImage** **fpp_image, **VRmImageFormat** f_image_format, **VRmBYTE** *fp_buffer, **VRmDWORD** f_pitch)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetFrameCounter** (const **VRmImage** *fcp_image, **VRmDWORD** *fp_frame_counter)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetImageFooter** (const **VRmImage** *fcp_image, const void **fpp_data, **VRmDWORD** *fp_size)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetImageBufferSize** (const **VRmImage** *fcp_image, **VRmDWORD** *fp_size)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSetImageBufferSize** (const **VRmImage** *fcp_image, **VRmDWORD** f_size)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetImageSensorPort** (const **VRmImage** *fcp_image, **VRmDWORD** *fp_port)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamFreeImage** (**VRmImage** **fpp_image)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetTargetFormatListSize** (const **VRmImageFormat** *fcp_source_format, **VRmDWORD** *fp_size)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetTargetFormatListEntry** (const **VRmImageFormat** *fcp_source_format, **VRmDWORD** f_index, **VRmImageFormat** *fp_target_format)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamConvertImage** (const **VRmImage** *fcp_source, **VRmImage** *fp_target)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetStringFromColorFormat** (VRmColorFormat f_color_format, **VRmSTRING** *fp_string)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPixelDepthFromColorFormat** (VRmColorFormat f_color_format, **VRmDWORD** *fp_pixel_depth)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamCompareImageFormats** (const **VRmImageFormat** *fcp_format1, const **VRmImageFormat** *fcp_format2, **VRmBOOL** *fp_result)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetImageLut** (const **VRmImage** *fcp_image, const **VRmBYTE** **fpp_lut, **VRmDWORD** *fp_size)

Variables

- VRmColorFormat **m_color_format**

- int **m_image_modifier**
- **VRmImageFormat**
- **VRmBYTE * mp_buffer**
- **VRmDWORD m_pitch**
- double **m_time_stamp**
- **VRmImage**

Detailed Description

Macro Definition Documentation

#define VRM_IMAGE_BUFFER_SIZE_MAX (0xFFFFFFFF)

special value for **VRmUsbCamSetImageBufferSize()**

Typedef Documentation

typedef VRM_ENUM _VRmColorFormat

enum for color formats

typedef VRM_STRUCT _VRmImage

```
Initial value:{

        VRmImageFormat m_image_format
```

struct for image container

typedef VRM_STRUCT _VRmImageFormat

```
Initial value:{

        VRmDWORD m_width
```

struct for image format

typedef VRM_ENUM _VRmImageModifier

enum for image modifiers

Function Documentation

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamCompareImageFormats (const VRmImageFormat * *fcp_format1*, const VRmImageFormat * *fcp_format2*, VRmBOOL * *fp_result*)

compare two image formats.

sets target of fp_result to 0 if keys are equal, sets it to 1 otherwise

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamConvertImage (const VRmImage * *fcp_source*, VRmImage * *fp_target*)

convert source image to target image.

use **VRmUsbCamGetTargetFormatListSize()** and **VRmUsbCamGetTargetFormatListEntry()** to generally find out about possible conversions for an image with format fcp_source->m_image_format. use **VRmUsbCamGetTargetFormatListSizeEx2()** and **VRmUsbCamGetTargetFormatListEntryEx2()** instead, when you like to convert a source image acquired from the device. this takes additional converter settings of the device into account.

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamCopyImage (VRmImage ** *fpp_image*, const VRmImage * *fcp_src_image*)

create new image as copy of src_image.

NOTE: free image using **VRmUsbCamFreeImage()**.

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamCropImage (VRmImage ** *fpp_image*, const VRmImage * *fcp_src_image*, const VRmRectI * *fcp_roi*)

get cropped Part of an Image without copying of image data.

NOTE: image data is shared with given image. NOTE: free image using **VRmUsbCamFreeImage()**.

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamFreeImage (VRmImage ** *fpp_image*)

free VRmImage created by **VRmUsbCamNewImage()**, **VRmUsbCamCopyImage()** or **VRmUsbCamSetImage()**

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetFrameCounter (const VRmImage * *fcp_image*, VRmDWORD * *fp_frame_counter*)

get Frame Counter of Image

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetImageBufferSize (const VRmImage * *fcp_image*, VRmDWORD * *fp_size*)

get Size of Image Buffer (mp_buffer)

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetImageFooter (const VRmImage * *fcp_image*, const void ** *fpp_data*, VRmDWORD * *fp_size*)

get Footer Data of Image.

NOTE: this is only valid as long as the given image is valid.

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetImageLut (const VRmImage * *fcp_image*, const VRmBYTE ** *fpp_lut*, VRmDWORD * *fp_size*)

get the LUT that is associated with the given VRmImage

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetImageSensorPort (const VRmImage * *fcp_image*, VRmDWORD * *fp_port*)

get the Image Sensor Port this image originates from.

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetPixelDepthFromColorFormat (VRmColorFormat *f_color_format*, VRmDWORD * *fp_pixel_depth*)

get pixel depth (in bytes) from color format

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetStringFromColorFormat (VRmColorFormat *f_color_format*, VRmSTRING * *fp_string*)

get string representation from color format

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetTargetFormatListEntry (const VRmImageFormat * *fcp_source_format*, VRmDWORD *f_index*, VRmImageFormat * *fp_target_format*)

query target format list entry with index = [0...size-1]

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetTargetFormatListSize (const VRmImageFormat * *fcp_source_format*, VRmDWORD * *fp_size*)

get number of available target format list entries for a given source format

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamNewImage (VRmImage ** *fpp_image*, VRmImageFormat *f_image_format*)

create new image with given image format.

NOTE: free image using **VRmUsbCamFreeImage()**.

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamSetImage (VRmImage ** *fpp_image*, VRmImageFormat *f_image_format*, VRmBYTE * *fp_buffer*, VRmDWORD *f_pitch*)

create image container with given format, buffer and pitch.

NOTE: free image using **VRmUsbCamFreeImage()** but you still have to deallocate fp_buffer YOURSELF!

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamSetImageBufferSize (const VRmImage * *fcp_image*, VRmDWORD *f_size*)

set Size of Image Buffer (mp_buffer).

NOTE: in general, this is only applicable for RLE image buffers. resetting the size to VRM_IMAGE_BUFFER_SIZE_MAX resets the size to the maximum value allowable for the format of the image.

Variable Documentation

VRmColorFormat m_color_format

color format enum

int m_image_modifier

bit combination of enum VRmImageModifier values

VRmDWORD m_pitch

pitch = number of bytes from (x, y) to (x, y+1)

double m_time_stamp

timestamp of image in ms since last **VRmUsbCamRestartTimer()**

VRmBYTE* mp_buffer

pointer to image buffer

VRmImage

VRmImageFormat

Frame Grabber

Functions

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetSensorPortListSize** (**VRmUsbCamDevice** f_device, **VRmDWORD** *fp_size)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetSensorPortListEntry** (**VRmUsbCamDevice** f_device, **VRmDWORD** f_index, **VRmDWORD** *fp_port)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetSourceFormatEx** (**VRmUsbCamDevice** f_device, **VRmDWORD** f_port, **VRmImageFormat** *fp_source_format)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetSourceFormatDescription** (**VRmUsbCamDevice** f_device, **VRmDWORD** f_port, **VRmSTRING** *fp_string)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamFindSensorPortListIndex** (**VRmUsbCamDevice** f_device, **VRmDWORD** f_port, **VRmDWORD** *fp_index)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetTargetFormatListSizeEx2** (**VRmUsbCamDevice** f_device, **VRmDWORD** f_port, **VRmDWORD** *fp_size)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetTargetFormatListEntryEx2** (**VRmUsbCamDevice** f_device, **VRmDWORD** f_port, **VRmDWORD** f_index, **VRmImageFormat** *fp_target_format)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamStart** (**VRmUsbCamDevice** f_device)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamStop** (**VRmUsbCamDevice** f_device)

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetRunning** (**VRmUsbCamDevice** f_device, **VRmBOOL** *fp_running)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamResetFrameCounter** (**VRmUsbCamDevice** f_device)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamIsNextImageReadyEx** (**VRmUsbCamDevice** f_device, **VRmDWORD** f_port, **VRmBOOL** *fp_ready)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamLockNextImageEx2** (**VRmUsbCamDevice** f_device, **VRmDWORD** f_port, **VRmImage** **fpp_image, **VRmDWORD** *fp_frames_dropped, int f_timeout_ms)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamLockNextImageEx** (**VRmUsbCamDevice** f_device, **VRmDWORD** f_port, **VRmImage** **fpp_image, **VRmDWORD** *fp_frames_dropped)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamUnlockNextImage** (**VRmUsbCamDevice** f_device, **VRmImage** **fpp_image)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSoftTrigger** (**VRmUsbCamDevice** f_device)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetDeviceLut** (**VRmUsbCamDevice** f_device, **VRmDWORD** f_port, const **VRmBYTE** **fpp_lut, **VRmDWORD** *fp_size, **VRmDWORD** *fp_format_modifier)

Detailed Description

Function Documentation

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamFindSensorPortListIndex (VRmUsbCamDevice *f_device*, VRmDWORD *f_port*, VRmDWORD * *fp_index*)

utility function: query index in sensor port list for specified sensor port number

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetDeviceLut (VRmUsbCamDevice *f_device*, VRmDWORD *f_port*, const VRmBYTE ** *fpp_lut*, VRmDWORD * *fp_size*, VRmDWORD * *fp_format_modifier*)

get the LUT that is associated with a specific sensor port.

the format modifier returned is a bit combination of VRM_CORRECTION_LUT_xxx constants.

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetRunning (VRmUsbCamDevice *f_device*, VRmBOOL * *fp_running*)

check if frame grabber is running

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetSensorPortListEntry (VRmUsbCamDevice *f_device*, VRmDWORD *f_index*, VRmDWORD * *fp_port*)

query image sensor port number entry with index [0...size-1]

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetSensorPortListSize (VRmUsbCamDevice *f_device*, VRmDWORD * *fp_size*)

get number of image sensor ports

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetSourceFormatDescription (VRmUsbCamDevice *f_device*, VRmDWORD *f_port*, VRmSTRING * *fp_string*)

get string description of source format NOTE: single-sensor devices do only have port #1

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetSourceFormatEx (VRmUsbCamDevice *f_device*, VRmDWORD *f_port*, VRmImageFormat * *fp_source_format*)

get current source format of given sensor port NOTE: single-sensor devices do only have port #1

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetTargetFormatListEntryEx2 (VRmUsbCamDevice *f_device*, VRmDWORD *f_port*, VRmDWORD *f_index*, VRmImageFormat * *fp_target_format*)

query target format list entry at given sensor port with index = [0...size-1]

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetTargetFormatListSizeEx2 (VRmUsbCamDevice *f_device*, VRmDWORD *f_port*, VRmDWORD * *fp_size*)

get number of available target format list entries for current source format at given sensor port NOTE: single-sensor devices do only have port #1

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamIsNextImageReadyEx (VRmUsbCamDevice *f_device*, VRmDWORD *f_port*, VRmBOOL * *fp_ready*)

is next image ready.

check if the next image from specified sensor port can be immediately accessed via VRmUsbCamLockNextImage(). if not, you can do something else and check again. specify a port of #0 if the port number doesn't matter. NOTE: single-sensor devices do only have port #1

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamLockNextImageEx (VRmUsbCamDevice *f_device*, VRmDWORD *f_port*, VRmImage ** *fpp_image*, VRmDWORD * *fp_frames_dropped*)

obsolete API function to get a locked source image.

this function is equivalent to **VRmUsbCamLockNextImageEx2()** with f_timeout_ms=0. WARNING: this function will never return, if no image becomes ready and will potentially stall your application indefinitely.

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamLockNextImageEx2 (VRmUsbCamDevice *f_device*, VRmDWORD *f_port*, VRmImage ** *fpp_image*, VRmDWORD * *fp_frames_dropped*, int *f_timeout_ms*)

get a locked source image from a specified sensor port.

this functions returns with value VRM_FAIL, if no image is locked within approximately f_timeout_ms milliseconds. use f_timeout_ms=0 to wait indefinitely for a locked image (latency optimal, but dangerous, and only applicable for freerunning/constantly triggered applications) use fp_frames_dropped to see how many frames have been dropped before this image (optional, pass NULL if drops don't matter to you). specify a port of #0 if the port number doesn't matter. use **VRmUsbCamConvertImage()** to convert source image into a traget image. NOTE: single-sensor devices do only have port #1

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamResetFrameCounter (VRmUsbCamDevice *f_device*)

reset frame counter

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamSoftTrigger (VRmUsbCamDevice *f_device*)

soft trigger.

initiate a soft trigger if the current device supports this, otherwise the function fails

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamStart (VRmUsbCamDevice *f_device*)

start frame grabber.

allocates image buffers (queue size) and starts the first image transfers

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamStop (VRmUsbCamDevice *f_device*)

terminates frame grabber, frees unlocked buffers, locked buffers are kept until released with VRmUsbCamUnlockNextImage

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamUnlockNextImage (VRmUsbCamDevice *f_device*, VRmImage ** *fpp_image*)

unlock source image.

you have to Unlock the image which was locked via VRmUsbCamLockNextImage(), do NOT use **VRmUsbCamFreeImage()** to free this image. NOTE: if the image has successfully been unlocked, *fpp_image will be set to NULL. nevertheless, the function may return an error when re-queuing the buffer into the grabber ring buffer failed.

Configuration Settings

Typedefs

- typedef unsigned int **VRmUsbCamConfigID**

Functions

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamLoadConfig** (**VRmUsbCamDevice** f_device, **VRmUsbCamConfigID** f_id)

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSaveConfig** (**VRmUsbCamDevice** f_device, **VRmUsbCamConfigID** f_id)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSaveConfigRequiresFirmwareCompression** (**VRmUsbCamDevice** f_device, **VRmUsbCamConfigID** f_id, **VRmBOOL** *fp_required)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamDeleteConfig** (**VRmUsbCamDevice** f_device, **VRmUsbCamConfigID** f_id)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetConfigData** (**VRmUsbCamDevice** f_device, **VRmUserData** **fpp_data)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSetConfigData** (**VRmUsbCamDevice** f_device, const **VRmUserData** *fcp_data)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamConfigIncludesUnsupportedValues** (**VRmUsbCamDevice** f_device, **VRmBOOL** *fp_value)

Detailed Description

Typedef Documentation

typedef unsigned int VRmUsbCamConfigID

config id.

there are several possible camera configs stored in hardware: VRmUsbCamConfigID = 0 is the factory default config (read-only) VRmUsbCamConfigID = 1 is the user default config (first user config, automatically loaded at **VRmUsbCamOpenDevice()**) VRmUsbCamConfigID = 2 is the second user config ... VRmUsbCamConfigID = 9 is the last user config

---

Function Documentation

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamConfigIncludesUnsupportedValues (VRmUsbCamDevice *f_device*, VRmBOOL * *fp_value*)

returns true if **VRmUsbCamSaveConfig()** / **VRmUsbCamGetConfigData()** may drop some values.

this happens if the config was initially created by a newer application and contains settings that are not supported by this software version and that therefore will be discarded

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamDeleteConfig (VRmUsbCamDevice *f_device*, VRmUsbCamConfigID *f_id*)

deletes the given config from the device. valid values for f_id = 2 to 9

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetConfigData (VRmUsbCamDevice *f_device*, VRmUserData ** *fpp_data*)

gets current device config en bloque as user data.

use **VRmUsbCamFreeUserData()** to free the returned buffer after usage

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamLoadConfig (VRmUsbCamDevice *f_device*, VRmUsbCamConfigID *f_id*)

load config.

valid values for f_id = 0 to 9 NOTE: the grabber must be stopped when you call this function.

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamSaveConfig (VRmUsbCamDevice *f_device*, VRmUsbCamConfigID *f_id*)

saves the current config in hardware.

valid values for f_id = 1 to 9 NOTE: this might take some seconds (blocking) in case of an necessary firmware compression, use VRmUsbCamSaveConfigRequiresFirmwareCompression to check if this will happen.

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamSaveConfigRequiresFirmwareCompression (VRmUsbCamDevice *f_device*, VRmUsbCamConfigID *f_id*, VRmBOOL * *fp_required*)

check if next **VRmUsbCamSaveConfig()** requires a firmware compression. valid values for f_id = 1 to 9.

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamSetConfigData (VRmUsbCamDevice *f_device*, const VRmUserData * *fcp_data*)

sets current device config en bloque from user data.

activate config data that was previously returned by VRmUsbCamGetConfig

Property Management

Modules

- **Properties of type VRM_PROP_TYPE_BOOL**
- **Properties of type VRM_PROP_TYPE_INT**
- **Properties of type VRM_PROP_TYPE_FLOAT**
- **Properties of type VRM_PROP_TYPE_DOUBLE**
- **Properties of type VRM_PROP_TYPE_STRING.**
- **Properties of type VRM_PROP_TYPE_ENUM**
- **Properties of type VRM_PROP_TYPE_SIZE_I**
- **Properties of type VRM_PROP_TYPE_POINT_I**
- **Properties of type VRM_PROP_TYPE_RECT_I**

Typedefs

- typedef **VRM_ENUM _VRmPropType**
- typedef **VRM_STRUCT _VRmPropInfo**

Functions

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyListSize** (**VRmUsbCamDevice** f_device, **VRmDWORD** *fp_size)

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyListEntry** (**VRmUsbCamDevice** f_device, **VRmDWORD** f_index, VRmPropId *fp_id)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyInfo** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmPropInfo** *fp_info)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertySupported** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmBOOL** *fp_supported)

Variables

- VRmPropType **m_type**
- **VRmSTRING m_id_string**
- **VRmSTRING m_description**
- **VRmBOOL m_writeable**
- **VRmPropInfo**

Detailed Description

Typedef Documentation

typedef VRM_STRUCT _VRmPropInfo

```
Initial value:{
        VRmPropId   m_id
```

general information about a property.

see **VRmUsbCamGetPropertyInfo()** NOTE: the strings within this struct are only guaranteed to be valid until the next API call!

typedef VRM_ENUM _VRmPropType

```
Initial value:{
        VRM_PROP_TYPE_BOOL = 1,
        VRM_PROP_TYPE_INT,
        VRM_PROP_TYPE_FLOAT,
        VRM_PROP_TYPE_STRING,
        VRM_PROP_TYPE_ENUM,
        VRM_PROP_TYPE_SIZE_I,
        VRM_PROP_TYPE_POINT_I,
        VRM_PROP_TYPE_RECT_I,
        VRM_PROP_TYPE_DOUBLE
} VRmPropType
```

enumeration of supported property value types.

see struct returned from **VRmUsbCamGetPropertyInfo()**

Function Documentation

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetPropertyInfo (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, VRmPropInfo * *fp_info*)

get info struct of property

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetPropertyListEntry (VRmUsbCamDevice *f_device*, VRmDWORD *f_index*, VRmPropId * *fp_id*)

get identifier of property with index [0...size-1]

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetPropertyListSize (VRmUsbCamDevice *f_device*, VRmDWORD * *fp_size*)

get number of available properties

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetPropertySupported (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, VRmBOOL * *fp_supported*)

check whether some specific property is supported

Variable Documentation

VRmSTRING m_description

VRmSTRING m_id_string

VRmPropType m_type

VRmBOOL m_writeable

VRmPropInfo

Misc Functions

Functions

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSavePNG** (**VRmSTRING** fcp_file_name, const **VRmImage** *fcp_image, int f_z_compression_level)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamLoadPNG** (**VRmSTRING** fcp_file_name, **VRmImage** **fpp_image)

Detailed Description

Function Documentation

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamLoadPNG (VRmSTRING *fcp_file_name*, VRmImage ** *fpp_image*)

Load a VRmImage from a PNG file.

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamSavePNG (VRmSTRING *fcp_file_name*, const VRmImage * *fcp_image*, int *f_z_compression_level*)

Save a VRmImage to a PNG file. Compression level 0 is uncompressed, max. is 9, use default with -1.

Callbacks/Events

Typedefs

- typedef **VRM_ENUM _VRmStaticCallbackType**
- typedef **VRM_ENUM _VRmDeviceChangeType**
- typedef **VRM_STRUCT _VRmStaticCallbackCMemAllocationChangeParams**
- typedef void(**VRM_API** * **VRmStaticCallback**)(VRmStaticCallbackType f_type, void *fp_user_data, const void *fcp_callback_params)
- typedef **VRM_ENUM _VRmDeviceCallbackType**
- typedef void(**VRM_API** * **VRmDeviceCallback**)(**VRmUsbCamDevice** f_device, VRmDeviceCallbackType f_type, void *fp_user_data, const void *fcp_callback_params)

Functions

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamRegisterStaticCallback** (**VRmStaticCallback** fp_callback, void *fp_user_data)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamUnregisterStaticCallback** (**VRmStaticCallback** fp_callback)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamRegisterStaticCallbackEx** (**VRmStaticCallback** fp_callback, void *fp_user_data)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamUnregisterStaticCallbackEx** (**VRmStaticCallback** fp_callback, void *fp_user_data)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamRegisterDeviceCallbackEx** (**VRmUsbCamDevice** f_device, **VRmDeviceCallback** fp_callback, void *fp_user_data)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamUnregisterDeviceCallbackEx** (**VRmUsbCamDevice** f_device, **VRmDeviceCallback** fp_callback, void *fp_user_data)

Variables

- void * **mp_virtual**
- void * **mp_physical**
- **VRmDWORD m_size**
- **VRmStaticCallbackCMemAllocationChangeParams**

Detailed Description

Typedef Documentation

typedef VRM_ENUM _VRmDeviceCallbackType

```
Initial value:{
```

```
        VRM_DEVICE_CALLBACK_TYPE_LUT_CHANGED = 1,

        VRM_DEVICE_CALLBACK_TYPE_SOURCE_FORMAT_CHANGED = 2,

        VRM_DEVICE_CALLBACK_TYPE_PROPERTY_VALUE_CHANGED = 3,

        VRM_DEVICE_CALLBACK_TYPE_PROPERTY_LIST_CHANGED = 4,

        VRM_DEVICE_CALLBACK_TYPE_SOURCE_FORMAT_LIST_CHANGED = 5,

        VRM_DEVICE_CALLBACK_TYPE_TARGET_FORMAT_LIST_CHANGED = 6,

        VRM_DEVICE_CALLBACK_TYPE_PROPERTY_INFO_CHANGED = 7,

        VRM_DEVICE_CALLBACK_TYPE_PROPERTY_ATTRIBS_CHANGED = 8
} VRmDeviceCallbackType
```

Device Callback Types.

see **VRmUsbCamRegisterDeviceCallbackEx()**. these callbacks may be called synchronously from within any VRmUsbCamXXX() functions!

typedef VRM_ENUM _VRmDeviceChangeType

```
Initial value:{

        VRM_DEVICE_CHANGE_TYPE_ARRIVAL = 1,

        VRM_DEVICE_CHANGE_TYPE_REMOVECOMPLETE = 2,

        VRM_DEVICE_CHANGE_TYPE_BUSY = 3
} VRmDeviceChangeType
```

Parameter definition for VRM_STATIC_CALLBACK_TYPE_DEVICE_CHANGE callback.

typedef VRM_STRUCT _VRmStaticCallbackCMemAllocationChangeParams

```
Initial value:{

        VRmBOOL  m_allocate
```

parameter definition for VRM_STATIC_CALLBACK_TYPE_CMEM_ALLOCATION_CHANGE callback

typedef VRM_ENUM _VRmStaticCallbackType

```
Initial value:{

        VRM_STATIC_CALLBACK_TYPE_DEVICE_CHANGE = 1,

        VRM_STATIC_CALLBACK_TYPE_CMEM_ALLOCATION_CHANGE = 2
} VRmStaticCallbackType
```

VRmUsbCam (Static) Callback Types.

see **VRmUsbCamRegisterStaticCallback()**.

typedef void(VRM_API * VRmDeviceCallback)(VRmUsbCamDevice f_device,VRmDeviceCallbackType f_type,void *fp_user_data,const void *fcp_callback_params)

Callback function signature definition for **VRmUsbCamRegisterDeviceCallbackEx()**

typedef void(VRM_API * VRmStaticCallback)(VRmStaticCallbackType f_type,void *fp_user_data,const void *fcp_callback_params)

Callback function signature definition for **VRmUsbCamRegisterStaticCallback()**

Function Documentation

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamRegisterDeviceCallbackEx (VRmUsbCamDevice *f_device*, VRmDeviceCallback *fp_callback*, void * *fp_user_data*)

register a callback function for the given device.

NOTE: in contrast to VRmUsbCamRegisterDeviceCallback(), this functions allows registering a specific callback function pointer multiple times per device as long as fp_user_data is different each time. you should use **VRmUsbCamUnregisterDeviceCallbackEx()** for later unregistration specifying the same function pointer and fp_user_data pair as used for registration.

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamRegisterStaticCallback (VRmStaticCallback *fp_callback*, void * *fp_user_data*)

register a static callback function.

NOTE: a specific callback function pointer can only be registered once per device.

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamRegisterStaticCallbackEx (VRmStaticCallback *fp_callback*, void * *fp_user_data*)

register a static callback function.

NOTE: in contrast to **VRmUsbCamRegisterStaticCallback()**, this functions allows registering a specific callback function pointer multiple times as long as fp_user_data is different each time. you should use **VRmUsbCamUnregisterStaticCallbackEx()** for later unregistration specifying the same function pointer and fp_user_data pair as used for registration.

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamUnregisterDeviceCallbackEx (VRmUsbCamDevice *f_device*, VRmDeviceCallback *fp_callback*, void * *fp_user_data*)

unregister a callback function for the given device.

specify the same function pointer and fp_user_data as used at registration.

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamUnregisterStaticCallback (VRmStaticCallback *fp_callback*)

unregister a static callback function

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamUnregisterStaticCallbackEx (VRmStaticCallback *fp_callback*, void * *fp_user_data*)

unregister a static callback function.

specify the same function pointer and fp_user_data as used at registration.

Variable Documentation

VRmDWORD m_size

size (in bytes) of memory area

void* mp_physical

physical address of memory area

void* mp_virtual

virtual address of memory area

VRmStaticCallbackCMemAllocationChangeParams

VM_LIB related functions

Functions

- **VRM_EXTERN** VRmRetVal **VRM_API VRmCreateVMLIBKey** (**VRmDWORD** *fp_vmlib_key)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmCreateVMLIBKeyDsp** (**VRmDWORD** *fp_vmlib_key)

Detailed Description

Function Documentation

VRM_EXTERN VRmRetVal VRM_API VRmCreateVMLIBKey (VRmDWORD * *fp_vmlib_key*)

VRM_EXTERN VRmRetVal VRM_API VRmCreateVMLIBKeyDsp (VRmDWORD * *fp_vmlib_key*)

Device Property Page Support Functions

Typedefs

- typedef **VRM_STRUCT _VRmDevicePropertyPage**

Functions

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamCreateDevicePropertyPage** (**VRmUsbCamDevice** f_device, HWND f_hwnd_parent, const **VRmRectI** *fp_rectangle, **VRmDevicePropertyPage** **fpp_page)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamDestroyDevicePropertyPage** (**VRmDevicePropertyPage** **fpp_page)

Variables

- **VRmSizeI m_size_hint**
- void * **mp_private**
- **VRmDevicePropertyPage**

Detailed Description

Typedef Documentation

typedef VRM_STRUCT _VRmDevicePropertyPage

```
Initial value:{

        HWND m_handle
```

represents a device property page (device GUI).

use **VRmUsbCamCreateDevicePropertyPage()** to create it and destroy it afterwards with **VRmUsbCamDestroyDevicePropertyPage()**. NOTE: you may move, resize, hide/show the page with Windows API calls using the m_handle Window Handle

---

Function Documentation

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamCreateDevicePropertyPage (VRmUsbCamDevice *f_device*, HWND *f_hwnd_parent*, const VRmRectI * *fp_rectangle*, VRmDevicePropertyPage ** *fpp_page*)

create a property page for the device.

if f_hwnd_parent is null, a new independent window is created, if f_hwnd_parent is not null, a child window is created within the given parent. fp_rectangle may optionally point to a rectangle containing the initial window position. NOTE: in case a new independent window is created, it is not shown automatically, so you need to show the window by

```
ShowWindow((*fpp_page)->m_handle, SW_SHOW)
```

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamDestroyDevicePropertyPage (VRmDevicePropertyPage ** *fpp_page*)

destroy the given property page

Variable Documentation

VRmSizeI m_size_hint

the size this page looks best

void* mp_private

API private informations.

VRmDevicePropertyPage

Advanced Device Management

Functions

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamActivateDevice** (**VRmWORD** f_vendor_id, **VRmWORD** f_product_id, **VRmDWORD** f_serial)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamDeactivateDevice** (**VRmWORD** f_vendor_id, **VRmWORD** f_product_id, **VRmDWORD** f_serial)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamActivateAllDevices** (void)

Detailed Description

Function Documentation

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamActivateAllDevices (void)

activates all VRmagic USB devices in the current hardware profile that are currently deactivated

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamActivateDevice (VRmWORD *f_vendor_id*, VRmWORD *f_product_id*, VRmDWORD *f_serial*)

activates a VRmagic USB device in the current hardware profile, identified by vendor id, product id and serial number

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamDeactivateDevice (VRmWORD *f_vendor_id*, VRmWORD *f_product_id*, VRmDWORD *f_serial*)

deactivates a VRmagic USB device in the current hardware profile, identified by vendor id, product id and serial number.

deactivates devices no longer consume USB bandwidth and are marked by a red cross in the device manager. they will also not be included in the device key list returned by **VRmUsbCamGetDeviceKeyListSize()**/VRmUsbCamGetDeviceKeyListSize()! NOTE: deactivated devices remain deactivated until explicit re-activation, even after replug!

Properties of type VRM_PROP_TYPE_BOOL

Typedefs

- typedef **VRM_STRUCT _VRmPropAttribsB**

Functions

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyValueB** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmBOOL** *fp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSetPropertyValueB** (**VRmUsbCamDevice** f_device, VRmPropId f_id, const **VRmBOOL** *fcp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyAttribsB** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmPropAttribsB** *fp_attribs)

Variables

- **VRmBOOL m_min**
- **VRmBOOL m_max**
- **VRmBOOL m_step**
- **VRmPropAttribsB**

Detailed Description

Typedef Documentation

typedef VRM_STRUCT _VRmPropAttribsB

```
Initial value:{
        VRmBOOL m_default
```

Function Documentation

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetPropertyAttribsB (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, VRmPropAttribsB * *fp_attribs*)

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetPropertyValueB (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, VRmBOOL * *fp_value*)

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamSetPropertyValueB (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, const VRmBOOL * *fcp_value*)

Variable Documentation

VRmRectI m_max

VRmRectI m_min

VRmRectI m_step

VRmPropAttribsB

Properties of type VRM_PROP_TYPE_INT

Typedefs

- typedef **VRM_STRUCT _VRmPropAttribsI**

Functions

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyValueI** (**VRmUsbCamDevice** f_device, VRmPropId f_id, int *fp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSetPropertyValueI** (**VRmUsbCamDevice** f_device, VRmPropId f_id, const int *fcp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyAttribsI** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmPropAttribsI** *fp_attribs)

Variables

- **VRmPropAttribsI**

Detailed Description

Typedef Documentation

typedef VRM_STRUCT _VRmPropAttribsI

```
Initial value:{
        int m_default
```

Function Documentation

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetPropertyAttribsI (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, VRmPropAttribsI * *fp_attribs*)

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetPropertyValueI (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, int * *fp_value*)

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamSetPropertyValueI (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, const int * *fcp_value*)

Variable Documentation

VRmPropAttribsI

Properties of type VRM_PROP_TYPE_FLOAT

Typedefs

- typedef **VRM_STRUCT _VRmPropAttribsF**

Functions

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyValueF** (**VRmUsbCamDevice** f_device, VRmPropId f_id, float *fp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSetPropertyValueF** (**VRmUsbCamDevice** f_device, VRmPropId f_id, const float *fcp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyAttribsF** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmPropAttribsF** *fp_attribs)

Variables

- **VRmPropAttribsF**

Detailed Description

Typedef Documentation

typedef VRM_STRUCT _VRmPropAttribsF

```
Initial value:{
        float m_default
```

Function Documentation

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetPropertyAttribsF (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, VRmPropAttribsF * *fp_attribs*)

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetPropertyValueF (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, float * *fp_value*)

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamSetPropertyValueF (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, const float * *fcp_value*)

Variable Documentation

VRmPropAttribsF

Properties of type VRM_PROP_TYPE_DOUBLE

Typedefs

- typedef **VRM_STRUCT _VRmPropAttribsD**

Functions

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyValueD** (**VRmUsbCamDevice** f_device, VRmPropId f_id, double *fp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSetPropertyValueD** (**VRmUsbCamDevice** f_device, VRmPropId f_id, const double *fcp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyAttribsD** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmPropAttribsD** *fp_attribs)

Variables

- **VRmPropAttribsD**

Detailed Description

Typedef Documentation

typedef VRM_STRUCT _VRmPropAttribsD

```
Initial value:{
        double m_default
```

Function Documentation

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetPropertyAttribsD (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, VRmPropAttribsD * *fp_attribs*)

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetPropertyValueD (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, double * *fp_value*)

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamSetPropertyValueD (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, const double * *fcp_value*)

Variable Documentation

VRmPropAttribsD

Properties of type VRM_PROP_TYPE_STRING.

Typedefs

- typedef **VRM_STRUCT _VRmPropAttribsS**

Functions

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyValueS** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmSTRING** *fp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSetPropertyValueS** (**VRmUsbCamDevice** f_device, VRmPropId f_id, const **VRmSTRING** *fcp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyAttribsS** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmPropAttribsS** *fp_attribs)

Variables

- **VRmPropAttribsS**

Detailed Description

NOTE: the strings within the VRmPropAttribsS struct are only guaranteed to be valid until the next API call!

NOTE: the string returned by **VRmUsbCamGetPropertyValueS()** is only guaranteed to be valid until the next API call!

Typedef Documentation

typedef VRM_STRUCT _VRmPropAttribsS

```
Initial value:{
        VRmSTRING m_default
```

Function Documentation

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetPropertyAttribsS (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, VRmPropAttribsS * *fp_attribs*)

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetPropertyValueS (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, VRmSTRING * *fp_value*)

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamSetPropertyValueS (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, const VRmSTRING * *fcp_value*)

Variable Documentation

VRmPropAttribsS

Properties of type VRM_PROP_TYPE_ENUM

Typedefs

- typedef **VRM_STRUCT _VRmPropAttribsE**

Functions

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyValueE** (**VRmUsbCamDevice** f_device, VRmPropId f_id, VRmPropId *fp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSetPropertyValueE** (**VRmUsbCamDevice** f_device, VRmPropId f_id, const VRmPropId *fcp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyAttribsE** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmPropAttribsE** *fp_attribs)

Variables

- **VRmPropAttribsE**

Detailed Description

Typedef Documentation

typedef VRM_STRUCT _VRmPropAttribsE

```
Initial value:{
        VRmPropId m_default
```

Function Documentation

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetPropertyAttribsE (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, VRmPropAttribsE * *fp_attribs*)

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetPropertyValueE (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, VRmPropId * *fp_value*)

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamSetPropertyValueE (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, const VRmPropId * *fcp_value*)

Variable Documentation

VRmPropAttribsE

Properties of type VRM_PROP_TYPE_SIZE_I

Typedefs

- typedef **VRM_STRUCT _VRmPropAttribsSizeI**

Functions

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyValueSizeI** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmSizeI** *fp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSetPropertyValueSizeI** (**VRmUsbCamDevice** f_device, VRmPropId f_id, const **VRmSizeI** *fcp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyAttribsSizeI** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmPropAttribsSizeI** *fp_attribs)

Variables

- **VRmPropAttribsSizeI**

Detailed Description

Typedef Documentation

typedef VRM_STRUCT _VRmPropAttribsSizeI

```
Initial value:{
        VRmSizeI m_default
```

Function Documentation

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetPropertyAttribsSizeI (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, VRmPropAttribsSizeI * *fp_attribs*)

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetPropertyValueSizeI (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, VRmSizeI * *fp_value*)

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamSetPropertyValueSizeI (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, const VRmSizeI * *fcp_value*)

Variable Documentation

VRmPropAttribsSizeI

Properties of type VRM_PROP_TYPE_POINT_I

Typedefs

- typedef **VRM_STRUCT _VRmPropAttribsPointI**

Functions

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyValuePointI** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmPointI** *fp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSetPropertyValuePointI** (**VRmUsbCamDevice** f_device, VRmPropId f_id, const **VRmPointI** *fcp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyAttribsPointI** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmPropAttribsPointI** *fp_attribs)

Variables

- **VRmPropAttribsPointI**

Detailed Description

Typedef Documentation

typedef VRM_STRUCT _VRmPropAttribsPointI

```
Initial value:{
        VRmPointI m_default
```

Function Documentation

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetPropertyAttribsPointI (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, VRmPropAttribsPointI * *fp_attribs*)

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetPropertyValuePointI (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, VRmPointI * *fp_value*)

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamSetPropertyValuePointI (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, const VRmPointI * *fcp_value*)

Variable Documentation

VRmPropAttribsPointI

Properties of type VRM_PROP_TYPE_RECT_I

Typedefs

- typedef **VRM_STRUCT _VRmPropAttribsRectI**

Functions

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyValueRectI** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmRectI** *fp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSetPropertyValueRectI** (**VRmUsbCamDevice** f_device, VRmPropId f_id, const **VRmRectI** *fcp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyAttribsRectI** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmPropAttribsRectI** *fp_attribs)

Variables

- **VRmPropAttribsRectI**

Detailed Description

Typedef Documentation

typedef VRM_STRUCT _VRmPropAttribsRectI

```
Initial value:{
        VRmRectI m_default
```

Function Documentation

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetPropertyAttribsRectI (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, VRmPropAttribsRectI * *fp_attribs*)

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamGetPropertyValueRectI (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, VRmRectI * *fp_value*)

VRM_EXTERN VRmRetVal VRM_API VRmUsbCamSetPropertyValueRectI (VRmUsbCamDevice *f_device*, VRmPropId *f_id*, const VRmRectI * *fcp_value*)

Variable Documentation

VRmPropAttribsRectI

File Documentation

vrmusbcam2.h File Reference

Macros

- #define **VRMUSBCAM_VERSION** 3300
- #define **VRM_API**
- #define **VRM_EXTERN** extern
- #define **VRM_STRUCT** struct
- #define **VRM_ENUM** enum
- #define **VRM_VRMUSBCAMDEVICE_DEFINED**
- #define **VRM_IMAGE_BUFFER_SIZE_MAX** (0xFFFFFFFF)

Typedefs

- typedef unsigned char **VRmBYTE**
- typedef unsigned short int **VRmWORD**
- typedef unsigned int **VRmDWORD**
- typedef unsigned int **VRmBOOL**
- typedef const char * **VRmSTRING**
- typedef **VRM_STRUCT _VRmSizeI**
- typedef **VRM_STRUCT _VRmPointI**
- typedef **VRM_STRUCT _VRmRectI**
- typedef **VRM_ENUM _VRmRetVal**
- typedef **VRM_ENUM _VRmErrorCode**
- typedef struct
- VRmUsbCamDeviceInternal * **VRmUsbCamDevice**
- typedef **VRM_STRUCT _VRmDeviceKey**
- typedef **VRM_STRUCT _VRmUserData**
- typedef **VRM_ENUM _VRmColorFormat**
- typedef **VRM_ENUM _VRmImageModifier**
- typedef **VRM_STRUCT _VRmImageFormat**
- typedef **VRM_STRUCT _VRmImage**
- typedef unsigned int **VRmUsbCamConfigID**

- typedef **VRM_ENUM _VRmPropType**
- typedef **VRM_STRUCT _VRmPropInfo**
- typedef **VRM_STRUCT _VRmPropAttribsB**
- typedef **VRM_STRUCT _VRmPropAttribsI**
- typedef **VRM_STRUCT _VRmPropAttribsF**
- typedef **VRM_STRUCT _VRmPropAttribsD**
- typedef **VRM_STRUCT _VRmPropAttribsS**
- typedef **VRM_STRUCT _VRmPropAttribsE**
- typedef **VRM_STRUCT _VRmPropAttribsSizeI**
- typedef **VRM_STRUCT _VRmPropAttribsPointI**
- typedef **VRM_STRUCT _VRmPropAttribsRectI**
- typedef **VRM_ENUM _VRmStaticCallbackType**
- typedef **VRM_ENUM _VRmDeviceChangeType**
- typedef **VRM_STRUCT _VRmStaticCallbackCMemAllocationChangeParams**
- typedef void(**VRM_API** * **VRmStaticCallback**)(VRmStaticCallbackType f_type, void *fp_user_data, const void *fcp_callback_params)
- typedef **VRM_ENUM _VRmDeviceCallbackType**
- typedef void(**VRM_API** * **VRmDeviceCallback**)(**VRmUsbCamDevice** f_device, VRmDeviceCallbackType f_type, void *fp_user_data, const void *fcp_callback_params)

Functions

- **VRM_EXTERN VRmSTRING VRM_API VRmUsbCamGetLastError** (void)
- **VRM_EXTERN** int **VRM_API VRmUsbCamGetLastErrorCode** (void)
- **VRM_EXTERN** void **VRM_API VRmUsbCamClearLastError** (void)
- **VRM_EXTERN VRmBOOL VRM_API VRmUsbCamLastErrorWasTriggerTimeout** (void)
- **VRM_EXTERN VRmBOOL VRM_API VRmUsbCamLastErrorWasTriggerStall** (void)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamEnableLogging** (void)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamEnableLoggingEx** (**VRmSTRING** f_log_file_name)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetVersion** (**VRmDWORD** *fp_version)
- **VRM_EXTERN** void **VRM_API VRmUsbCamCleanup** (void)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamUpdateDeviceKeyList** (void)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamUpdateDeviceKeyListEx** (**VRmBOOL** f_local, **VRmBOOL** f_usb, **VRmBOOL** f_ethernet)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetDeviceKeyListSize** (**VRmDWORD** *fp_size)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetDeviceKeyListEntry** (**VRmDWORD** f_index, **VRmDeviceKey** **fpp_device_key)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetVendorId** (const **VRmDeviceKey** *fcp_device_key, **VRmWORD** *fp_vendor_id)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetProductId** (const **VRmDeviceKey** *fcp_device_key, **VRmWORD** *fp_product_id)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetGroupId** (const **VRmDeviceKey** *fcp_device_key, **VRmWORD** *fp_group_id)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetSerialString** (const **VRmDeviceKey** *fcp_device_key, **VRmSTRING** *fp_serial_str)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetIpAddress** (const **VRmDeviceKey** *fcp_device_key, **VRmSTRING** *fp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetLocalIpAddress** (const **VRmDeviceKey** *fcp_device_key, **VRmSTRING** *fp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamCompareDeviceKeys** (const **VRmDeviceKey** *fcp_device_key1, const **VRmDeviceKey** *fcp_device_key2, **VRmBOOL** *fp_result)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamFreeDeviceKey** (**VRmDeviceKey** **fpp_device_key)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamOpenDevice** (const **VRmDeviceKey** *fcp_device_key, **VRmUsbCamDevice** *fp_device)

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetDeviceKey** (**VRmUsbCamDevice** f_device, **VRmDeviceKey** **fpp_device_key)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamCloseDevice** (**VRmUsbCamDevice** f_device)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamLoadUserData** (**VRmUsbCamDevice** f_device, **VRmUserData** **fpp_user_data)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSaveUserData** (**VRmUsbCamDevice** f_device, const **VRmUserData** *fcp_user_data)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamNewUserData** (**VRmUserData** **fpp_user_data, **VRmDWORD** f_length)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamFreeUserData** (**VRmUserData** **fpp_user_data)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamRestartTimer** (void)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetCurrentTime** (double *fp_current_time)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamNewImage** (**VRmImage** **fpp_image, **VRmImageFormat** f_image_format)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamCopyImage** (**VRmImage** **fpp_image, const **VRmImage** *fcp_src_image)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamCropImage** (**VRmImage** **fpp_image, const **VRmImage** *fcp_src_image, const **VRmRectI** *fcp_roi)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSetImage** (**VRmImage** **fpp_image, **VRmImageFormat** f_image_format, **VRmBYTE** *fp_buffer, **VRmDWORD** f_pitch)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetFrameCounter** (const **VRmImage** *fcp_image, **VRmDWORD** *fp_frame_counter)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetImageFooter** (const **VRmImage** *fcp_image, const void **fpp_data, **VRmDWORD** *fp_size)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetImageBufferSize** (const **VRmImage** *fcp_image, **VRmDWORD** *fp_size)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSetImageBufferSize** (const **VRmImage** *fcp_image, **VRmDWORD** f_size)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetImageSensorPort** (const **VRmImage** *fcp_image, **VRmDWORD** *fp_port)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamFreeImage** (**VRmImage** **fpp_image)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetTargetFormatListSize** (const **VRmImageFormat** *fcp_source_format, **VRmDWORD** *fp_size)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetTargetFormatListEntry** (const **VRmImageFormat** *fcp_source_format, **VRmDWORD** f_index, **VRmImageFormat** *fp_target_format)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamConvertImage** (const **VRmImage** *fcp_source, **VRmImage** *fp_target)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetStringFromColorFormat** (VRmColorFormat f_color_format, **VRmSTRING** *fp_string)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPixelDepthFromColorFormat** (VRmColorFormat f_color_format, **VRmDWORD** *fp_pixel_depth)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamCompareImageFormats** (const **VRmImageFormat** *fcp_format1, const **VRmImageFormat** *fcp_format2, **VRmBOOL** *fp_result)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetImageLut** (const **VRmImage** *fcp_image, const **VRmBYTE** **fpp_lut, **VRmDWORD** *fp_size)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetSensorPortListSize** (**VRmUsbCamDevice** f_device, **VRmDWORD** *fp_size)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetSensorPortListEntry** (**VRmUsbCamDevice** f_device, **VRmDWORD** f_index, **VRmDWORD** *fp_port)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetSourceFormatEx** (**VRmUsbCamDevice** f_device, **VRmDWORD** f_port, **VRmImageFormat** *fp_source_format)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetSourceFormatDescription** (**VRmUsbCamDevice** f_device, **VRmDWORD** f_port, **VRmSTRING** *fp_string)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamFindSensorPortListIndex** (**VRmUsbCamDevice** f_device, **VRmDWORD** f_port, **VRmDWORD** *fp_index)

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetTargetFormatListSizeEx2** (**VRmUsbCamDevice** f_device, **VRmDWORD** f_port, **VRmDWORD** *fp_size)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetTargetFormatListEntryEx2** (**VRmUsbCamDevice** f_device, **VRmDWORD** f_port, **VRmDWORD** f_index, **VRmImageFormat** *fp_target_format)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamStart** (**VRmUsbCamDevice** f_device)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamStop** (**VRmUsbCamDevice** f_device)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetRunning** (**VRmUsbCamDevice** f_device, **VRmBOOL** *fp_running)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamResetFrameCounter** (**VRmUsbCamDevice** f_device)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamIsNextImageReadyEx** (**VRmUsbCamDevice** f_device, **VRmDWORD** f_port, **VRmBOOL** *fp_ready)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamLockNextImageEx2** (**VRmUsbCamDevice** f_device, **VRmDWORD** f_port, **VRmImage** **fpp_image, **VRmDWORD** *fp_frames_dropped, int f_timeout_ms)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamLockNextImageEx** (**VRmUsbCamDevice** f_device, **VRmDWORD** f_port, **VRmImage** **fpp_image, **VRmDWORD** *fp_frames_dropped)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamUnlockNextImage** (**VRmUsbCamDevice** f_device, **VRmImage** **fpp_image)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSoftTrigger** (**VRmUsbCamDevice** f_device)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetDeviceLut** (**VRmUsbCamDevice** f_device, **VRmDWORD** f_port, const **VRmBYTE** **fpp_lut, **VRmDWORD** *fp_size, **VRmDWORD** *fp_format_modifier)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamLoadConfig** (**VRmUsbCamDevice** f_device, **VRmUsbCamConfigID** f_id)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSaveConfig** (**VRmUsbCamDevice** f_device, **VRmUsbCamConfigID** f_id)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSaveConfigRequiresFirmwareCompression** (**VRmUsbCamDevice** f_device, **VRmUsbCamConfigID** f_id, **VRmBOOL** *fp_required)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamDeleteConfig** (**VRmUsbCamDevice** f_device, **VRmUsbCamConfigID** f_id)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetConfigData** (**VRmUsbCamDevice** f_device, **VRmUserData** **fpp_data)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSetConfigData** (**VRmUsbCamDevice** f_device, const **VRmUserData** *fcp_data)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamConfigIncludesUnsupportedValues** (**VRmUsbCamDevice** f_device, **VRmBOOL** *fp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyListSize** (**VRmUsbCamDevice** f_device, **VRmDWORD** *fp_size)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyListEntry** (**VRmUsbCamDevice** f_device, **VRmDWORD** f_index, VRmPropId *fp_id)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyInfo** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmPropInfo** *fp_info)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertySupported** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmBOOL** *fp_supported)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyValueB** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmBOOL** *fp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSetPropertyValueB** (**VRmUsbCamDevice** f_device, VRmPropId f_id, const **VRmBOOL** *fcp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyAttribsB** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmPropAttribsB** *fp_attribs)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyValueI** (**VRmUsbCamDevice** f_device, VRmPropId f_id, int *fp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSetPropertyValueI** (**VRmUsbCamDevice** f_device, VRmPropId f_id, const int *fcp_value)

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyAttribsI** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmPropAttribsI** *fp_attribs)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyValueF** (**VRmUsbCamDevice** f_device, VRmPropId f_id, float *fp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSetPropertyValueF** (**VRmUsbCamDevice** f_device, VRmPropId f_id, const float *fcp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyAttribsF** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmPropAttribsF** *fp_attribs)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyValueD** (**VRmUsbCamDevice** f_device, VRmPropId f_id, double *fp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSetPropertyValueD** (**VRmUsbCamDevice** f_device, VRmPropId f_id, const double *fcp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyAttribsD** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmPropAttribsD** *fp_attribs)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyValueS** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmSTRING** *fp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSetPropertyValueS** (**VRmUsbCamDevice** f_device, VRmPropId f_id, const **VRmSTRING** *fcp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyAttribsS** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmPropAttribsS** *fp_attribs)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyValueE** (**VRmUsbCamDevice** f_device, VRmPropId f_id, VRmPropId *fp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSetPropertyValueE** (**VRmUsbCamDevice** f_device, VRmPropId f_id, const VRmPropId *fcp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyAttribsE** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmPropAttribsE** *fp_attribs)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyValueSizeI** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmSizeI** *fp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSetPropertyValueSizeI** (**VRmUsbCamDevice** f_device, VRmPropId f_id, const **VRmSizeI** *fcp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyAttribsSizeI** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmPropAttribsSizeI** *fp_attribs)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyValuePointI** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmPointI** *fp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSetPropertyValuePointI** (**VRmUsbCamDevice** f_device, VRmPropId f_id, const **VRmPointI** *fcp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyAttribsPointI** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmPropAttribsPointI** *fp_attribs)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyValueRectI** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmRectI** *fp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSetPropertyValueRectI** (**VRmUsbCamDevice** f_device, VRmPropId f_id, const **VRmRectI** *fcp_value)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamGetPropertyAttribsRectI** (**VRmUsbCamDevice** f_device, VRmPropId f_id, **VRmPropAttribsRectI** *fp_attribs)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamSavePNG** (**VRmSTRING** fcp_file_name, const **VRmImage** *fcp_image, int f_z_compression_level)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamLoadPNG** (**VRmSTRING** fcp_file_name, **VRmImage** **fpp_image)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamRegisterStaticCallback** (**VRmStaticCallback** fp_callback, void *fp_user_data)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamUnregisterStaticCallback** (**VRmStaticCallback** fp_callback)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamRegisterStaticCallbackEx** (**VRmStaticCallback** fp_callback, void *fp_user_data)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamUnregisterStaticCallbackEx** (**VRmStaticCallback** fp_callback, void *fp_user_data)

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamRegisterDeviceCallbackEx** (**VRmUsbCamDevice** f_device, **VRmDeviceCallback** fp_callback, void *fp_user_data)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamUnregisterDeviceCallbackEx** (**VRmUsbCamDevice** f_device, **VRmDeviceCallback** fp_callback, void *fp_user_data)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmCreateVMLIBKey** (**VRmDWORD** *fp_vmlib_key)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmCreateVMLIBKeyDsp** (**VRmDWORD** *fp_vmlib_key)

Variables

- int **m_height**
- **VRmSizeI**
- int **m_y**
- **VRmPointI**
- int **m_top**
- int **m_width**
- **VRmRectI**
- **VRmSTRING mp_manufacturer_str**
- **VRmSTRING mp_product_str**
- **VRmBOOL m_busy**
- void * **mp_private**
- **VRmDeviceKey**
- **VRmBYTE** * **mp_data**
- **VRmUserData**
- VRmColorFormat **m_color_format**
- int **m_image_modifier**
- **VRmImageFormat**
- **VRmBYTE** * **mp_buffer**
- **VRmDWORD m_pitch**
- double **m_time_stamp**
- **VRmImage**
- VRmPropType **m_type**
- **VRmSTRING m_id_string**
- **VRmSTRING m_description**
- **VRmBOOL m_writeable**
- **VRmPropInfo**
- **VRmBOOL m_min**
- **VRmBOOL m_max**
- **VRmBOOL m_step**
- **VRmPropAttribsB**
- **VRmPropAttribsI**
- **VRmPropAttribsF**
- **VRmPropAttribsD**
- **VRmPropAttribsS**
- **VRmPropAttribsE**
- **VRmPropAttribsSizeI**
- **VRmPropAttribsPointI**
- **VRmPropAttribsRectI**
- void * **mp_virtual**
- void * **mp_physical**
- **VRmDWORD m_size**
- **VRmStaticCallbackCMemAllocationChangeParams**

Detailed Description

VRmUsbCam C API v3.3.0.0.

Macro Definition Documentation

#define VRM_API

VRmUsbCam is C calling convention (cdecl).

define this appropriate if your compiler has a different default

#define VRM_ENUM enum

#define VRM_EXTERN extern

#define VRM_STRUCT struct

#define VRMUSBCAM_VERSION 3300

version of this API header.

you can compare it with the value returned from **VRmUsbCamGetVersion()**

---

vrmusbcam2win32.h File Reference

Macros

- #define **VRM_API**
- #define **VRM_EXTERN** extern
- #define **VRM_STRUCT** struct

Typedefs

- typedef **VRM_STRUCT _VRmDevicePropertyPage**

Functions

- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamCreateDevicePropertyPage** (**VRmUsbCamDevice** f_device, HWND f_hwnd_parent, const **VRmRectI** *fp_rectangle, **VRmDevicePropertyPage** **fpp_page)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamDestroyDevicePropertyPage** (**VRmDevicePropertyPage** **fpp_page)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamActivateDevice** (**VRmWORD** f_vendor_id, **VRmWORD** f_product_id, **VRmDWORD** f_serial)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamDeactivateDevice** (**VRmWORD** f_vendor_id, **VRmWORD** f_product_id, **VRmDWORD** f_serial)
- **VRM_EXTERN** VRmRetVal **VRM_API VRmUsbCamActivateAllDevices** (void)

Variables

- **VRmSizeI m_size_hint**
- void * **mp_private**
- **VRmDevicePropertyPage**

Detailed Description

VRmUsbCam C API Win32-related Functions v3.3.0.0.

Macro Definition Documentation

#define VRM_API

VRmUsbCam is C calling convention (cdecl).

define this appropriate if your compiler has a different default

#define VRM_EXTERN extern

#define VRM_STRUCT struct

Index