NAME

Direct3D 9 Interoperability [DEPRECATED] −

Enumerations

enum **CUd3d9map_flags**
enum **CUd3d9register_flags**

Functions

CUresult cuD3D9MapResources (unsigned int count, IDirect3DResource9 **ppResource)
Map Direct3D resources for access by CUDA.
CUresult cuD3D9RegisterResource (IDirect3DResource9 *pResource, unsigned int Flags)
Register a Direct3D resource for access by CUDA.
CUresult cuD3D9ResourceGetMappedArray (**CUarray** *pArray, IDirect3DResource9 *pResource, unsigned int Face, unsigned int Level)
Get an array through which to access a subresource of a Direct3D resource which has been mapped for access by CUDA.
CUresult cuD3D9ResourceGetMappedPitch (size_t *pPitch, size_t *pPitchSlice, IDirect3DResource9 *pResource, unsigned int Face, unsigned int Level)
Get the pitch of a subresource of a Direct3D resource which has been mapped for access by CUDA.
CUresult cuD3D9ResourceGetMappedPointer (**CUdeviceptr** *pDevPtr, IDirect3DResource9 *pResource, unsigned int Face, unsigned int Level)
Get the pointer through which to access a subresource of a Direct3D resource which has been mapped for access by CUDA.
CUresult cuD3D9ResourceGetMappedSize (size_t *pSize, IDirect3DResource9 *pResource, unsigned int Face, unsigned int Level)
Get the size of a subresource of a Direct3D resource which has been mapped for access by CUDA.
CUresult cuD3D9ResourceGetSurfaceDimensions (size_t *pWidth, size_t *pHeight, size_t *pDepth, IDirect3DResource9 *pResource, unsigned int Face, unsigned int Level)
Get the dimensions of a registered surface.
CUresult cuD3D9ResourceSetMapFlags (IDirect3DResource9 *pResource, unsigned int Flags)
Set usage flags for mapping a Direct3D resource.
CUresult cuD3D9UnmapResources (unsigned int count, IDirect3DResource9 **ppResource)
Unmaps Direct3D resources.
CUresult cuD3D9UnregisterResource (IDirect3DResource9 *pResource)
Unregister a Direct3D resource.

Detailed Description

\brief deprecated Direct3D 9 interoperability functions of the low-level CUDA driver API (**cudaD3D9.h**)

This section describes deprecated Direct3D 9 interoperability functionality.

Enumeration Type Documentation

enum CUd3d9map_flags

Flags to map or unmap a resource

enum CUd3d9register_flags

Flags to register a resource

Function Documentation

CUresult cuD3D9MapResources (unsigned int count, IDirect3DResource9 ** ppResource)

Deprecated

This function is deprecated as of CUDA 3.0.

Maps the `count` Direct3D resources in `ppResource` for access by CUDA.

The resources in `ppResource` may be accessed in CUDA kernels until they are unmapped. Direct3D should not access any resources while they are mapped by CUDA. If an application does so the results are undefined.

This function provides the synchronization guarantee that any Direct3D calls issued before **cuD3D9MapResources()** will complete before any CUDA kernels issued after **cuD3D9MapResources()** begin.

If any of `ppResource` have not been registered for use with CUDA or if `ppResource` contains any duplicate entries, then **CUDA_ERROR_INVALID_HANDLE** is returned. If any of `ppResource` are presently mapped for access by CUDA, then **CUDA_ERROR_ALREADY_MAPPED** is returned.

Parameters:
count - Number of resources in ppResource
ppResource - Resources to map for CUDA usage

Returns:
CUDA_SUCCESS, **CUDA_ERROR_DEINITIALIZED**, **CUDA_ERROR_NOT_INITIALIZED**, **CUDA_ERROR_INVALID_CONTEXT**, **CUDA_ERROR_INVALID_HANDLE**, **CUDA_ERROR_ALREADY_MAPPED**, **CUDA_ERROR_UNKNOWN**

Note:
Note that this function may also return error codes from previous, asynchronous launches.

See also:
cuGraphicsMapResources

CUresult cuD3D9RegisterResource (IDirect3DResource9 * pResource, unsigned int Flags)

Deprecated
This function is deprecated as of CUDA 3.0.

Registers the Direct3D resource `pResource` for access by CUDA.

If this call is successful, then the application will be able to map and unmap this resource until it is unregistered through **cuD3D9UnregisterResource()**. Also on success, this call will increase the internal reference count on `pResource`. This reference count will be decremented when this resource is unregistered through **cuD3D9UnregisterResource()**.

This call is potentially high-overhead and should not be called every frame in interactive applications.

The type of `pResource` must be one of the following.

- IDirect3DVertexBuffer9: Cannot be used with `Flags` set to CU_D3D9_REGISTER_FLAGS_ARRAY.
- IDirect3DIndexBuffer9: Cannot be used with `Flags` set to CU_D3D9_REGISTER_FLAGS_ARRAY.
- IDirect3DSurface9: Only stand-alone objects of type IDirect3DSurface9 may be explicitly shared. In particular, individual mipmap levels and faces of cube maps may not be registered directly. To access individual surfaces associated with a texture, one must register the base texture object. For restrictions on the `Flags` parameter, see type IDirect3DBaseTexture9.
- IDirect3DBaseTexture9: When a texture is registered, all surfaces associated with the all mipmap levels of all faces of the texture will be accessible to CUDA.

The `Flags` argument specifies the mechanism through which CUDA will access the Direct3D resource. The following values are allowed.

- CU_D3D9_REGISTER_FLAGS_NONE: Specifies that CUDA will access this resource through a **CUdeviceptr**. The pointer, size, and (for textures), pitch for each subresource of this allocation may be queried through **cuD3D9ResourceGetMappedPointer()**, **cuD3D9ResourceGetMappedSize()**, and **cuD3D9ResourceGetMappedPitch()** respectively. This option is valid for all resource types.
- CU_D3D9_REGISTER_FLAGS_ARRAY: Specifies that CUDA will access this resource through a **CUarray** queried on a sub-resource basis through **cuD3D9ResourceGetMappedArray()**. This option is only valid for resources of type IDirect3DSurface9 and subtypes of IDirect3DBaseTexture9.

Not all Direct3D resources of the above types may be used for interoperability with CUDA. The following are some limitations.

- The primary rendertarget may not be registered with CUDA.
- Resources allocated as shared may not be registered with CUDA.
- Any resources allocated in D3DPOOL_SYSTEMMEM or D3DPOOL_MANAGED may not be registered with CUDA.

- Textures which are not of a format which is 1, 2, or 4 channels of 8, 16, or 32-bit integer or floating-point data cannot be shared.
- Surfaces of depth or stencil formats cannot be shared.

If Direct3D interoperability is not initialized on this context, then **CUDA_ERROR_INVALID_CONTEXT** is returned. If `pResource` is of incorrect type (e.g. is a non-stand-alone IDirect3DSurface9) or is already registered, then **CUDA_ERROR_INVALID_HANDLE** is returned. If `pResource` cannot be registered then **CUDA_ERROR_UNKNOWN** is returned.

Parameters:

pResource - Resource to register for CUDA access
Flags - Flags for resource registration

Returns:

CUDA_SUCCESS, **CUDA_ERROR_DEINITIALIZED**, **CUDA_ERROR_NOT_INITIALIZED**, **CUDA_ERROR_INVALID_CONTEXT**, **CUDA_ERROR_INVALID_VALUE**, **CUDA_ERROR_INVALID_HANDLE**, **CUDA_ERROR_OUT_OF_MEMORY**, **CUDA_ERROR_UNKNOWN**

Note:

Note that this function may also return error codes from previous, asynchronous launches.

See also:

cuGraphicsD3D9RegisterResource

CUresult cuD3D9ResourceGetMappedArray (CUarray * pArray, IDirect3DResource9 * pResource, unsigned int Face, unsigned int Level)

Deprecated

This function is deprecated as of CUDA 3.0.

Returns in `*pArray` an array through which the subresource of the mapped Direct3D resource `pResource` which corresponds to `Face` and `Level` may be accessed. The value set in `pArray` may change every time that `pResource` is mapped.

If `pResource` is not registered then **CUDA_ERROR_INVALID_HANDLE** is returned. If `pResource` was not registered with usage flags CU_D3D9_REGISTER_FLAGS_ARRAY then **CUDA_ERROR_INVALID_HANDLE** is returned. If `pResource` is not mapped then **CUDA_ERROR_NOT_MAPPED** is returned.

For usage requirements of `Face` and `Level` parameters, see **cuD3D9ResourceGetMappedPointer()**.

Parameters:

pArray - Returned array corresponding to subresource
pResource - Mapped resource to access
Face - Face of resource to access
Level - Level of resource to access

Returns:

CUDA_SUCCESS, **CUDA_ERROR_DEINITIALIZED**, **CUDA_ERROR_NOT_INITIALIZED**, **CUDA_ERROR_INVALID_CONTEXT**, **CUDA_ERROR_INVALID_VALUE**, **CUDA_ERROR_INVALID_HANDLE**, **CUDA_ERROR_NOT_MAPPED**

Note:

Note that this function may also return error codes from previous, asynchronous launches.

See also:

cuGraphicsSubResourceGetMappedArray

CUresult cuD3D9ResourceGetMappedPitch (size_t * pPitch, size_t * pPitchSlice, IDirect3DResource9 * pResource, unsigned int Face, unsigned int Level)

Deprecated

This function is deprecated as of CUDA 3.0.

Returns in `*pPitch` and `*pPitchSlice` the pitch and Z-slice pitch of the subresource of the mapped Direct3D resource `pResource`, which corresponds to `Face` and `Level`. The values set in `pPitch` and `pPitchSlice` may change every time that `pResource` is mapped.

The pitch and Z-slice pitch values may be used to compute the location of a sample on a surface as follows.

For a 2D surface, the byte offset of the sample at position **x**, **y** from the base pointer of the surface is:

y * **pitch** + (**bytes per pixel**) * **x**

For a 3D surface, the byte offset of the sample at position **x**, **y**, **z** from the base pointer of the surface is:

z* **slicePitch** + **y** * **pitch** + (**bytes per pixel**) * **x**

Both parameters `pPitch` and `pPitchSlice` are optional and may be set to NULL.

If `pResource` is not of type IDirect3DBaseTexture9 or one of its sub-types or if `pResource` has not been registered for use with CUDA, then cudaErrorInvalidResourceHandle is returned. If `pResource` was not registered with usage flags CU_D3D9_REGISTER_FLAGS_NONE, then **CUDA_ERROR_INVALID_HANDLE** is returned. If `pResource` is not mapped for access by CUDA then **CUDA_ERROR_NOT_MAPPED** is returned.

For usage requirements of `Face` and `Level` parameters, see **cuD3D9ResourceGetMappedPointer()**.

Parameters:

pPitch - Returned pitch of subresource
pPitchSlice - Returned Z-slice pitch of subresource
pResource - Mapped resource to access
Face - Face of resource to access
Level - Level of resource to access

Returns:

CUDA_SUCCESS, **CUDA_ERROR_DEINITIALIZED**, **CUDA_ERROR_NOT_INITIALIZED**, **CUDA_ERROR_INVALID_CONTEXT**, **CUDA_ERROR_INVALID_VALUE**, **CUDA_ERROR_INVALID_HANDLE**, **CUDA_ERROR_NOT_MAPPED**

Note:

Note that this function may also return error codes from previous, asynchronous launches.

See also:

cuGraphicsSubResourceGetMappedArray

CUresult cuD3D9ResourceGetMappedPointer (CUdeviceptr * pDevPtr, IDirect3DResource9 * pResource, unsigned int Face, unsigned int Level)

Deprecated

This function is deprecated as of CUDA 3.0.

Returns in `*pDevPtr` the base pointer of the subresource of the mapped Direct3D resource `pResource`, which corresponds to `Face` and `Level`. The value set in `pDevPtr` may change every time that `pResource` is mapped.

If `pResource` is not registered, then **CUDA_ERROR_INVALID_HANDLE** is returned. If `pResource` was not registered with usage flags CU_D3D9_REGISTER_FLAGS_NONE, then **CUDA_ERROR_INVALID_HANDLE** is returned. If `pResource` is not mapped, then **CUDA_ERROR_NOT_MAPPED** is returned.

If `pResource` is of type IDirect3DCubeTexture9, then `Face` must one of the values enumerated by type D3DCUBEMAP_FACES. For all other types `Face` must be 0. If `Face` is invalid, then **CUDA_ERROR_INVALID_VALUE** is returned.

If `pResource` is of type IDirect3DBaseTexture9, then `Level` must correspond to a valid mipmap level. At present only mipmap level 0 is supported. For all other types `Level` must be 0. If `Level` is invalid, then **CUDA_ERROR_INVALID_VALUE** is returned.

Parameters:

pDevPtr - Returned pointer corresponding to subresource
pResource - Mapped resource to access
Face - Face of resource to access
Level - Level of resource to access

Returns:

CUDA_SUCCESS, **CUDA_ERROR_DEINITIALIZED**, **CUDA_ERROR_NOT_INITIALIZED**, **CUDA_ERROR_INVALID_CONTEXT**, **CUDA_ERROR_INVALID_VALUE**, **CUDA_ERROR_INVALID_HANDLE**, **CUDA_ERROR_NOT_MAPPED**

Note:

Note that this function may also return error codes from previous, asynchronous launches.

See also:

cuGraphicsResourceGetMappedPointer

CUresult cuD3D9ResourceGetMappedSize (size_t * pSize, IDirect3DResource9 * pResource, unsigned int Face, unsigned int Level)

Deprecated

This function is deprecated as of CUDA 3.0.

Returns in `*pSize` the size of the subresource of the mapped Direct3D resource `pResource`, which corresponds to `Face` and `Level`. The value set in `pSize` may change every time that `pResource` is mapped.

If `pResource` has not been registered for use with CUDA, then **CUDA_ERROR_INVALID_HANDLE** is returned. If `pResource` was not registered with usage flags CU_D3D9_REGISTER_FLAGS_NONE, then **CUDA_ERROR_INVALID_HANDLE** is returned. If `pResource` is not mapped for access by CUDA, then **CUDA_ERROR_NOT_MAPPED** is returned.

For usage requirements of `Face` and `Level` parameters, see **cuD3D9ResourceGetMappedPointer**.

Parameters:

pSize - Returned size of subresource
pResource - Mapped resource to access
Face - Face of resource to access
Level - Level of resource to access

Returns:

CUDA_SUCCESS, **CUDA_ERROR_DEINITIALIZED**, **CUDA_ERROR_NOT_INITIALIZED**, **CUDA_ERROR_INVALID_CONTEXT**, **CUDA_ERROR_INVALID_VALUE**, **CUDA_ERROR_INVALID_HANDLE**, **CUDA_ERROR_NOT_MAPPED**

Note:

Note that this function may also return error codes from previous, asynchronous launches.

See also:

cuGraphicsResourceGetMappedPointer

CUresult cuD3D9ResourceGetSurfaceDimensions (size_t * pWidth, size_t * pHeight, size_t * pDepth, IDirect3DResource9 * pResource, unsigned int Face, unsigned int Level)

Deprecated

This function is deprecated as of CUDA 3.0.

Returns in `*pWidth`, `*pHeight`, and `*pDepth` the dimensions of the subresource of the mapped Direct3D resource `pResource`, which corresponds to `Face` and `Level`.

Because anti-aliased surfaces may have multiple samples per pixel, it is possible that the dimensions of a resource will be an integer factor larger than the dimensions reported by the Direct3D runtime.

The parameters `pWidth`, `pHeight`, and `pDepth` are optional. For 2D surfaces, the value returned in `*pDepth` will be 0.

If `pResource` is not of type IDirect3DBaseTexture9 or IDirect3DSurface9 or if `pResource` has not been registered for use with CUDA, then **CUDA_ERROR_INVALID_HANDLE** is returned.

For usage requirements of `Face` and `Level` parameters, see **cuD3D9ResourceGetMappedPointer()**.

Parameters:

pWidth - Returned width of surface
pHeight - Returned height of surface

pDepth - Returned depth of surface
pResource - Registered resource to access
Face - Face of resource to access
Level - Level of resource to access

Returns:
CUDA_SUCCESS, **CUDA_ERROR_DEINITIALIZED**, **CUDA_ERROR_NOT_INITIALIZED**, **CUDA_ERROR_INVALID_CONTEXT**, **CUDA_ERROR_INVALID_VALUE**, **CUDA_ERROR_INVALID_HANDLE**

Note:
Note that this function may also return error codes from previous, asynchronous launches.

See also:
cuGraphicsSubResourceGetMappedArray

CUresult cuD3D9ResourceSetMapFlags (IDirect3DResource9 * pResource, unsigned int Flags)

Deprecated
This function is deprecated as of Cuda 3.0.

Set `Flags` for mapping the Direct3D resource `pResource`.

Changes to `Flags` will take effect the next time `pResource` is mapped. The `Flags` argument may be any of the following:

- CU_D3D9_MAPRESOURCE_FLAGS_NONE: Specifies no hints about how this resource will be used. It is therefore assumed that this resource will be read from and written to by CUDA kernels. This is the default value.
- CU_D3D9_MAPRESOURCE_FLAGS_READONLY: Specifies that CUDA kernels which access this resource will not write to this resource.
- CU_D3D9_MAPRESOURCE_FLAGS_WRITEDISCARD: Specifies that CUDA kernels which access this resource will not read from this resource and will write over the entire contents of the resource, so none of the data previously stored in the resource will be preserved.

If `pResource` has not been registered for use with CUDA, then **CUDA_ERROR_INVALID_HANDLE** is returned. If `pResource` is presently mapped for access by CUDA, then **CUDA_ERROR_ALREADY_MAPPED** is returned.

Parameters:
pResource - Registered resource to set flags for
Flags - Parameters for resource mapping

Returns:
CUDA_SUCCESS, **CUDA_ERROR_DEINITIALIZED**, **CUDA_ERROR_NOT_INITIALIZED**, **CUDA_ERROR_INVALID_CONTEXT**, **CUDA_ERROR_INVALID_VALUE**, **CUDA_ERROR_INVALID_HANDLE**, **CUDA_ERROR_ALREADY_MAPPED**

Note:
Note that this function may also return error codes from previous, asynchronous launches.

See also:
cuGraphicsResourceSetMapFlags

CUresult cuD3D9UnmapResources (unsigned int count, IDirect3DResource9 ** ppResource)

Deprecated
This function is deprecated as of CUDA 3.0.

Unmaps the `count` Direct3D resources in `ppResource`.

This function provides the synchronization guarantee that any CUDA kernels issued before **cuD3D9UnmapResources()** will complete before any Direct3D calls issued after **cuD3D9UnmapResources()** begin.

If any of `ppResource` have not been registered for use with CUDA or if `ppResource` contains any duplicate entries, then **CUDA_ERROR_INVALID_HANDLE** is returned. If any of `ppResource` are not presently mapped for access by CUDA, then **CUDA_ERROR_NOT_MAPPED** is returned.

Parameters:
count - Number of resources to unmap for CUDA
ppResource - Resources to unmap for CUDA

Returns:
CUDA_SUCCESS, **CUDA_ERROR_DEINITIALIZED**, **CUDA_ERROR_NOT_INITIALIZED**, **CUDA_ERROR_INVALID_CONTEXT**, **CUDA_ERROR_INVALID_HANDLE**, **CUDA_ERROR_NOT_MAPPED**, **CUDA_ERROR_UNKNOWN**

Note:
Note that this function may also return error codes from previous, asynchronous launches.

See also:
cuGraphicsUnmapResources

CUresult cuD3D9UnregisterResource (IDirect3DResource9 * pResource)

Deprecated
This function is deprecated as of CUDA 3.0.

Unregisters the Direct3D resource `pResource` so it is not accessible by CUDA unless registered again.

If `pResource` is not registered, then **CUDA_ERROR_INVALID_HANDLE** is returned.

Parameters:
pResource - Resource to unregister

Returns:
CUDA_SUCCESS, **CUDA_ERROR_DEINITIALIZED**, **CUDA_ERROR_NOT_INITIALIZED**, **CUDA_ERROR_INVALID_CONTEXT**, **CUDA_ERROR_INVALID_HANDLE**, **CUDA_ERROR_UNKNOWN**

Note:
Note that this function may also return error codes from previous, asynchronous launches.

See also:
cuGraphicsUnregisterResource

Author

Generated automatically by Doxygen from the source code.