NAME

Texture Reference Management −

Functions

cudaError_t cudaBindTexture (size_t *offset, const struct **textureReference** *texref, const void *devPtr, const struct **cudaChannelFormatDesc** *desc, size_t size=UINT_MAX)
Binds a memory area to a texture.

cudaError_t cudaBindTexture2D (size_t *offset, const struct **textureReference** *texref, const void *devPtr, const struct **cudaChannelFormatDesc** *desc, size_t width, size_t height, size_t pitch)
Binds a 2D memory area to a texture.

cudaError_t cudaBindTextureToArray (const struct **textureReference** *texref, **cudaArray_const_t** array, const struct **cudaChannelFormatDesc** *desc)
Binds an array to a texture.

cudaError_t cudaBindTextureToMipmappedArray (const struct **textureReference** *texref, **cudaMipmappedArray_const_t** mipmappedArray, const struct **cudaChannelFormatDesc** *desc)
Binds a mipmapped array to a texture.

struct **cudaChannelFormatDesc cudaCreateChannelDesc** (int x, int y, int z, int w, enum **cudaChannelFormatKind** f)
Returns a channel descriptor using the specified format.

cudaError_t cudaGetChannelDesc (struct **cudaChannelFormatDesc** *desc, **cudaArray_const_t** array)
Get the channel descriptor of an array.

cudaError_t cudaGetTextureAlignmentOffset (size_t *offset, const struct **textureReference** *texref)
Get the alignment offset of a texture.

cudaError_t cudaGetTextureReference (const struct **textureReference** **texref, const void *symbol)
Get the texture reference associated with a symbol.

cudaError_t cudaUnbindTexture (const struct **textureReference** *texref)
Unbinds a texture.

Detailed Description

\brief texture reference management functions of the CUDA runtime API (cuda_runtime_api.h)

This section describes the low level texture reference management functions of the CUDA runtime application programming interface.

Some functions have overloaded C++ API template versions documented separately in the **C++ API Routines** module.

Function Documentation

cudaError_t cudaBindTexture (size_t * offset, const struct textureReference * texref, const void * devPtr, const struct cudaChannelFormatDesc * desc, size_t size = `UINT_MAX`**)**

Binds `size` bytes of the memory area pointed to by `devPtr` to the texture reference `texref`. `desc` describes how the memory is interpreted when fetching values from the texture. Any memory previously bound to `texref` is unbound.

Since the hardware enforces an alignment requirement on texture base addresses, **cudaBindTexture()** returns in `*offset` a byte offset that must be applied to texture fetches in order to read from the desired memory. This offset must be divided by the texel size and passed to kernels that read from the texture so they can be applied to the tex1Dfetch() function. If the device memory pointer was returned from **cudaMalloc()**, the offset is guaranteed to be 0 and NULL may be passed as the `offset` parameter.

The total number of elements (or texels) in the linear address range cannot exceed **cudaDeviceProp::maxTexture1DLinear**[0]. The number of elements is computed as (`size` / elementSize), where elementSize is determined from `desc`.

Parameters:

offset - Offset in bytes
texref - Texture to bind
devPtr - Memory area on device

desc - Channel format
size - Size of the memory area pointed to by devPtr

Returns:
cudaSuccess, **cudaErrorInvalidValue**, **cudaErrorInvalidDevicePointer**, **cudaErrorInvalidTexture**

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

See also:
cudaCreateChannelDesc (C API), **cudaGetChannelDesc**, **cudaGetTextureReference**, **cudaBindTexture (C++ API)**, **cudaBindTexture2D (C API)**, **cudaBindTextureToArray (C API)**, **cudaUnbindTexture (C API)**, **cudaGetTextureAlignmentOffset (C API)**

cudaError_t cudaBindTexture2D (size_t * offset, const struct textureReference * texref, const void * devPtr, const struct cudaChannelFormatDesc * desc, size_t width, size_t height, size_t pitch)
Binds the 2D memory area pointed to by `devPtr` to the texture reference `texref`. The size of the area is constrained by `width` in texel units, `height` in texel units, and `pitch` in byte units. `desc` describes how the memory is interpreted when fetching values from the texture. Any memory previously bound to `texref` is unbound.

Since the hardware enforces an alignment requirement on texture base addresses, **cudaBindTexture2D()** returns in `*offset` a byte offset that must be applied to texture fetches in order to read from the desired memory. This offset must be divided by the texel size and passed to kernels that read from the texture so they can be applied to the tex2D() function. If the device memory pointer was returned from **cudaMalloc()**, the offset is guaranteed to be 0 and NULL may be passed as the `offset` parameter.

`width` and `height`, which are specified in elements (or texels), cannot exceed **cudaDeviceProp::maxTexture2DLinear**[0] and **cudaDeviceProp::maxTexture2DLinear**[1] respectively. `pitch`, which is specified in bytes, cannot exceed **cudaDeviceProp::maxTexture2DLinear**[2].

The driver returns **cudaErrorInvalidValue** if `pitch` is not a multiple of **cudaDeviceProp::texturePitchAlignment**.

Parameters:
offset - Offset in bytes
texref - Texture reference to bind
devPtr - 2D memory area on device
desc - Channel format
width - Width in texel units
height - Height in texel units
pitch - Pitch in bytes

Returns:
cudaSuccess, **cudaErrorInvalidValue**, **cudaErrorInvalidDevicePointer**, **cudaErrorInvalidTexture**

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

See also:
cudaCreateChannelDesc (C API), **cudaGetChannelDesc**, **cudaGetTextureReference**, **cudaBindTexture (C API)**, **cudaBindTexture2D (C++ API)**, **cudaBindTexture2D (C++ API, inherited channel descriptor)**, **cudaBindTextureToArray (C API)**, **cudaBindTextureToArray (C API)**, **cudaGetTextureAlignmentOffset (C API)**

cudaError_t cudaBindTextureToArray (const struct textureReference * texref, cudaArray_const_t array, const struct cudaChannelFormatDesc * desc)
Binds the CUDA array `array` to the texture reference `texref`. `desc` describes how the memory is interpreted when fetching values from the texture. Any CUDA array previously bound to `texref` is unbound.

Parameters:

texref - Texture to bind
array - Memory array on device
desc - Channel format

Returns:
cudaSuccess, **cudaErrorInvalidValue**, **cudaErrorInvalidDevicePointer**, **cudaErrorInvalidTexture**

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

See also:
cudaCreateChannelDesc (C API), **cudaGetChannelDesc**, **cudaGetTextureReference**, **cudaBindTexture (C API)**, **cudaBindTexture2D (C API)**, **cudaBindTextureToArray (C++ API)**, **cudaUnbindTexture (C API)**, **cudaGetTextureAlignmentOffset (C API)**

cudaError_t cudaBindTextureToMipmappedArray (const struct textureReference * texref, cudaMipmappedArray_const_t mipmappedArray, const struct cudaChannelFormatDesc * desc)
Binds the CUDA mipmapped array `mipmappedArray` to the texture reference `texref`. `desc` describes how the memory is interpreted when fetching values from the texture. Any CUDA mipmapped array previously bound to `texref` is unbound.

Parameters:
texref - Texture to bind
mipmappedArray - Memory mipmapped array on device
desc - Channel format

Returns:
cudaSuccess, **cudaErrorInvalidValue**, **cudaErrorInvalidDevicePointer**, **cudaErrorInvalidTexture**

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

See also:
cudaCreateChannelDesc (C API), **cudaGetChannelDesc**, **cudaGetTextureReference**, **cudaBindTexture (C API)**, **cudaBindTexture2D (C API)**, **cudaBindTextureToArray (C++ API)**, **cudaUnbindTexture (C API)**, **cudaGetTextureAlignmentOffset (C API)**

struct cudaChannelFormatDesc cudaCreateChannelDesc (int x, int y, int z, int w, enum cudaChannelFormatKind f) `[read]`
Returns a channel descriptor with format `f` and number of bits of each component `x`, `y`, `z`, and `w`. The **cudaChannelFormatDesc** is defined as:

```
struct cudaChannelFormatDesc {
  int x, y, z, w;
  enum cudaChannelFormatKind f;
};
```

where **cudaChannelFormatKind** is one of **cudaChannelFormatKindSigned**, **cudaChannelFormatKindUnsigned**, or **cudaChannelFormatKindFloat**.

Parameters:
x - X component
y - Y component
z - Z component
w - W component
f - Channel format

Returns:
Channel descriptor with format `f`

See also:
cudaCreateChannelDesc (C++ API), **cudaGetChannelDesc**, **cudaGetTextureReference**, **cudaBindTexture (C API)**, **cudaBindTexture2D (C API)**, **cudaBindTextureToArray (C API)**, **cudaUnbindTexture (C API)**, **cudaGetTextureAlignmentOffset (C API)**

cudaError_t cudaGetChannelDesc (struct cudaChannelFormatDesc * desc, cudaArray_const_t array)
Returns in `*desc` the channel descriptor of the CUDA array `array`.

Parameters:
desc - Channel format
array - Memory array on device

Returns:
cudaSuccess, **cudaErrorInvalidValue**

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

See also:
cudaCreateChannelDesc (C API), **cudaGetTextureReference**, **cudaBindTexture (C API)**, **cudaBindTexture2D (C API)**, **cudaBindTextureToArray (C API)**, **cudaUnbindTexture (C API)**, **cudaGetTextureAlignmentOffset (C API)**

cudaError_t cudaGetTextureAlignmentOffset (size_t * offset, const struct textureReference * texref)
Returns in `*offset` the offset that was returned when texture reference `texref` was bound.

Parameters:
offset - Offset of texture reference in bytes
texref - Texture to get offset of

Returns:
cudaSuccess, **cudaErrorInvalidTexture**, **cudaErrorInvalidTextureBinding**

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

See also:
cudaCreateChannelDesc (C API), **cudaGetChannelDesc**, **cudaGetTextureReference**, **cudaBindTexture (C API)**, **cudaBindTexture2D (C API)**, **cudaBindTextureToArray (C API)**, **cudaUnbindTexture (C API)**, **cudaGetTextureAlignmentOffset (C++ API)**

cudaError_t cudaGetTextureReference (const struct textureReference ** texref, const void * symbol)
Returns in `*texref` the structure associated to the texture reference defined by symbol `symbol`.

Parameters:
texref - Texture reference associated with symbol
symbol - Texture to get reference for

Returns:
cudaSuccess, **cudaErrorInvalidTexture**

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

Use of a string naming a variable as the `symbol` parameter was removed in CUDA 5.0.

See also:
cudaCreateChannelDesc (C API), **cudaGetChannelDesc**, **cudaGetTextureAlignmentOffset (C API)**, **cudaBindTexture (C API)**, **cudaBindTexture2D (C API)**, **cudaBindTextureToArray (C API)**, **cudaUnbindTexture (C API)**

cudaError_t cudaUnbindTexture (const struct textureReference * texref)
Unbinds the texture bound to `texref`.

Parameters:
texref - Texture to unbind

Returns:
cudaSuccess

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

See also:
cudaCreateChannelDesc (C API), **cudaGetChannelDesc**, **cudaGetTextureReference**, **cudaBindTexture (C API)**, **cudaBindTexture2D (C API)**, **cudaBindTextureToArray (C API)**,

cudaUnbindTexture (C++ API), **cudaGetTextureAlignmentOffset (C API)**

Author

Generated automatically by Doxygen from the source code.