NVIDIA OptiX 7.5

API Reference Manual

17 May 2022

Version 7.5

Contents

1 NVIDIA OptiX 7.5 API

This document describes the OptiX 7.5 application programming interface. See https://raytracing-docs.nvidia.com/ for more information about programming with OptiX 7.5.

2 Module Index

2.1 Modules

Here is a list of all modules:

3 Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

4 Module Documentation

4.1 Device API

Functions

- template<typename... Payload>

  static __forceinline__
  __device__ void optixTrace (OptixTraversableHandle handle, float3 rayOrigin, float3 rayDirection, float tmin, float tmax, float rayTime, OptixVisibilityMask visibilityMask, unsigned int rayFlags, unsigned int SBToffset, unsigned int SBTstride, unsigned int missSBTIndex, Payload &...payload)
- template<typename... Payload>
  static __forceinline__
  __device__ void optixTrace (OptixPayloadTypeID type, OptixTraversableHandle handle, float3 rayOrigin, float3 rayDirection, float tmin, float tmax, float rayTime, OptixVisibilityMask visibilityMask, unsigned int rayFlags, unsigned int SBToffset, unsigned int SBTstride, unsigned int missSBTIndex, Payload &...payload)
- static __forceinline__
  __device__ void optixSetPayload_0 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_1 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_2 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_3 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_4 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_5 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_6 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_7 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_8 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_9 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_10 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_11 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_12 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_13 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_14 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_15 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_16 (unsigned int p)

- static __forceinline__ __device__ void optixSetPayload_17 (unsigned int p)
- static __forceinline__ __device__ void optixSetPayload_18 (unsigned int p)
- static __forceinline__ __device__ void optixSetPayload_19 (unsigned int p)
- static __forceinline__ __device__ void optixSetPayload_20 (unsigned int p)
- static __forceinline__ __device__ void optixSetPayload_21 (unsigned int p)
- static __forceinline__ __device__ void optixSetPayload_22 (unsigned int p)
- static __forceinline__ __device__ void optixSetPayload_23 (unsigned int p)
- static __forceinline__ __device__ void optixSetPayload_24 (unsigned int p)
- static __forceinline__ __device__ void optixSetPayload_25 (unsigned int p)
- static __forceinline__ __device__ void optixSetPayload_26 (unsigned int p)
- static __forceinline__ __device__ void optixSetPayload_27 (unsigned int p)
- static __forceinline__ __device__ void optixSetPayload_28 (unsigned int p)
- static __forceinline__ __device__ void optixSetPayload_29 (unsigned int p)
- static __forceinline__ __device__ void optixSetPayload_30 (unsigned int p)
- static __forceinline__ __device__ void optixSetPayload_31 (unsigned int p)
- static __forceinline__ __device__ unsigned int optixGetPayload_0 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_1 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_2 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_3 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_4 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_5 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_6 ()

- static __forceinline__ __device__ unsigned int optixGetPayload_7 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_8 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_9 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_10 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_11 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_12 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_13 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_14 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_15 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_16 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_17 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_18 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_19 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_20 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_21 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_22 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_23 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_24 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_25 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_26 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_27 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_28 ()

- static __forceinline__
  __device__ unsigned int optixGetPayload_29 ()
- static __forceinline__
  __device__ unsigned int optixGetPayload_30 ()
- static __forceinline__
  __device__ unsigned int optixGetPayload_31 ()
- static __forceinline__
  __device__ void optixSetPayloadTypes (unsigned int typeMask)
- static __forceinline__
  __device__ unsigned int optixUndefinedValue ()
- static __forceinline__
  __device__ float3 optixGetWorldRayOrigin ()
- static __forceinline__
  __device__ float3 optixGetWorldRayDirection ()
- static __forceinline__
  __device__ float3 optixGetObjectRayOrigin ()
- static __forceinline__
  __device__ float3 optixGetObjectRayDirection ()
- static __forceinline__
  __device__ float optixGetRayTmin ()
- static __forceinline__
  __device__ float optixGetRayTmax ()
- static __forceinline__
  __device__ float optixGetRayTime ()
- static __forceinline__
  __device__ unsigned int optixGetRayFlags ()
- static __forceinline__
  __device__ unsigned int optixGetRayVisibilityMask ()
- static __forceinline__
  __device__
  OptixTraversableHandle optixGetInstanceTraversableFromIAS (OptixTraversableHandle ias, unsigned int instIdx)
- static __forceinline__
  __device__ void optixGetTriangleVertexData (OptixTraversableHandle gas, unsigned int primIdx, unsigned int sbtGASIndex, float time, float3 data[3])
- static __forceinline__
  __device__ void optixGetLinearCurveVertexData (OptixTraversableHandle gas, unsigned int primIdx, unsigned int sbtGASIndex, float time, float4 data[2])
- static __forceinline__
  __device__ void optixGetQuadraticBSplineVertexData (OptixTraversableHandle gas, unsigned int primIdx, unsigned int sbtGASIndex, float time, float4 data[3])
- static __forceinline__
  __device__ void optixGetCubicBSplineVertexData (OptixTraversableHandle gas, unsigned int primIdx, unsigned int sbtGASIndex, float time, float4 data[4])

- static __forceinline__
  __device__ void optixGetCatmullRomVertexData (OptixTraversableHandle gas, unsigned int primIdx, unsigned int sbtGASIndex, float time, float4 data[4])
- static __forceinline__
  __device__ void optixGetSphereData (OptixTraversableHandle gas, unsigned int primIdx, unsigned int sbtGASIndex, float time, float4 data[1])
- static __forceinline__
  __device__
  OptixTraversableHandle optixGetGASTraversableHandle ()
- static __forceinline__
  __device__ float optixGetGASMotionTimeBegin (OptixTraversableHandle gas)
- static __forceinline__
  __device__ float optixGetGASMotionTimeEnd (OptixTraversableHandle gas)
- static __forceinline__
  __device__ unsigned int optixGetGASMotionStepCount (OptixTraversableHandle gas)
- static __forceinline__
  __device__ void optixGetWorldToObjectTransformMatrix (float m[12])
- static __forceinline__
  __device__ void optixGetObjectToWorldTransformMatrix (float m[12])
- static __forceinline__
  __device__ float3 optixTransformPointFromWorldToObjectSpace (float3 point)
- static __forceinline__
  __device__ float3 optixTransformVectorFromWorldToObjectSpace (float3 vec)
- static __forceinline__
  __device__ float3 optixTransformNormalFromWorldToObjectSpace (float3 normal)
- static __forceinline__
  __device__ float3 optixTransformPointFromObjectToWorldSpace (float3 point)
- static __forceinline__
  __device__ float3 optixTransformVectorFromObjectToWorldSpace (float3 vec)
- static __forceinline__
  __device__ float3 optixTransformNormalFromObjectToWorldSpace (float3 normal)
- static __forceinline__
  __device__ unsigned int optixGetTransformListSize ()
- static __forceinline__
  __device__
  OptixTraversableHandle optixGetTransformListHandle (unsigned int index)
- static __forceinline__
  __device__ OptixTransformType optixGetTransformTypeFromHandle (OptixTraversableHandle handle)
- static __forceinline__
  __device__ const
  OptixStaticTransform ∗ optixGetStaticTransformFromHandle (OptixTraversableHandle handle)
- static __forceinline__
  __device__ const
  OptixSRTMotionTransform ∗ optixGetSRTMotionTransformFromHandle (OptixTraversableHandle handle)

- static __forceinline__
  __device__ const
  OptixMatrixMotionTransform ∗ optixGetMatrixMotionTransformFromHandle
  (OptixTraversableHandle handle)
- static __forceinline__
  __device__ unsigned int optixGetInstanceIdFromHandle (OptixTraversableHandle handle)
- static __forceinline__
  __device__
  OptixTraversableHandle optixGetInstanceChildFromHandle (OptixTraversableHandle handle)
- static __forceinline__
  __device__ const float4 ∗ optixGetInstanceTransformFromHandle (OptixTraversableHandle
  handle)
- static __forceinline__
  __device__ const float4 ∗ optixGetInstanceInverseTransformFromHandle
  (OptixTraversableHandle handle)
- static __forceinline__
  __device__ bool optixReportIntersection (float hitT, unsigned int hitKind)
- static __forceinline__
  __device__ bool optixReportIntersection (float hitT, unsigned int hitKind, unsigned int a0)
- static __forceinline__
  __device__ bool optixReportIntersection (float hitT, unsigned int hitKind, unsigned int a0,
  unsigned int a1)
- static __forceinline__
  __device__ bool optixReportIntersection (float hitT, unsigned int hitKind, unsigned int a0,
  unsigned int a1, unsigned int a2)
- static __forceinline__
  __device__ bool optixReportIntersection (float hitT, unsigned int hitKind, unsigned int a0,
  unsigned int a1, unsigned int a2, unsigned int a3)
- static __forceinline__
  __device__ bool optixReportIntersection (float hitT, unsigned int hitKind, unsigned int a0,
  unsigned int a1, unsigned int a2, unsigned int a3, unsigned int a4)
- static __forceinline__
  __device__ bool optixReportIntersection (float hitT, unsigned int hitKind, unsigned int a0,
  unsigned int a1, unsigned int a2, unsigned int a3, unsigned int a4, unsigned int a5)
- static __forceinline__
  __device__ bool optixReportIntersection (float hitT, unsigned int hitKind, unsigned int a0,
  unsigned int a1, unsigned int a2, unsigned int a3, unsigned int a4, unsigned int a5, unsigned int
  a6)
- static __forceinline__
  __device__ bool optixReportIntersection (float hitT, unsigned int hitKind, unsigned int a0,
  unsigned int a1, unsigned int a2, unsigned int a3, unsigned int a4, unsigned int a5, unsigned int
  a6, unsigned int a7)
- static __forceinline__
  __device__ unsigned int optixGetAttribute_0 ()
- static __forceinline__
  __device__ unsigned int optixGetAttribute_1 ()

- static __forceinline__
  __device__ unsigned int optixGetAttribute_2 ()
- static __forceinline__
  __device__ unsigned int optixGetAttribute_3 ()
- static __forceinline__
  __device__ unsigned int optixGetAttribute_4 ()
- static __forceinline__
  __device__ unsigned int optixGetAttribute_5 ()
- static __forceinline__
  __device__ unsigned int optixGetAttribute_6 ()
- static __forceinline__
  __device__ unsigned int optixGetAttribute_7 ()
- static __forceinline__
  __device__ void optixTerminateRay ()
- static __forceinline__
  __device__ void optixIgnoreIntersection ()
- static __forceinline__
  __device__ unsigned int optixGetPrimitiveIndex ()
- static __forceinline__
  __device__ unsigned int optixGetSbtGASIndex ()
- static __forceinline__
  __device__ unsigned int optixGetInstanceId ()
- static __forceinline__
  __device__ unsigned int optixGetInstanceIndex ()
- static __forceinline__
  __device__ unsigned int optixGetHitKind ()
- static __forceinline__
  __device__ OptixPrimitiveType optixGetPrimitiveType (unsigned int hitKind)
- static __forceinline__
  __device__ bool optixIsFrontFaceHit (unsigned int hitKind)
- static __forceinline__
  __device__ bool optixIsBackFaceHit (unsigned int hitKind)
- static __forceinline__
  __device__ OptixPrimitiveType optixGetPrimitiveType ()
- static __forceinline__
  __device__ bool optixIsFrontFaceHit ()
- static __forceinline__
  __device__ bool optixIsBackFaceHit ()
- static __forceinline__
  __device__ bool optixIsTriangleHit ()
- static __forceinline__
  __device__ bool optixIsTriangleFrontFaceHit ()
- static __forceinline__
  __device__ bool optixIsTriangleBackFaceHit ()

- static __forceinline__
  __device__ float2 optixGetTriangleBarycentrics ()
- static __forceinline__
  __device__ float optixGetCurveParameter ()
- static __forceinline__
  __device__ uint3 optixGetLaunchIndex ()
- static __forceinline__
  __device__ uint3 optixGetLaunchDimensions ()
- static __forceinline__
  __device__ CUdeviceptr optixGetSbtDataPointer ()
- static __forceinline__
  __device__ void optixThrowException (int exceptionCode)
- static __forceinline__
  __device__ void optixThrowException (int exceptionCode, unsigned int exceptionDetail0)
- static __forceinline__
  __device__ void optixThrowException (int exceptionCode, unsigned int exceptionDetail0, unsigned int exceptionDetail1)
- static __forceinline__
  __device__ void optixThrowException (int exceptionCode, unsigned int exceptionDetail0, unsigned int exceptionDetail1, unsigned int exceptionDetail2)
- static __forceinline__
  __device__ void optixThrowException (int exceptionCode, unsigned int exceptionDetail0, unsigned int exceptionDetail1, unsigned int exceptionDetail2, unsigned int exceptionDetail3)
- static __forceinline__
  __device__ void optixThrowException (int exceptionCode, unsigned int exceptionDetail0, unsigned int exceptionDetail1, unsigned int exceptionDetail2, unsigned int exceptionDetail3, unsigned int exceptionDetail4)
- static __forceinline__
  __device__ void optixThrowException (int exceptionCode, unsigned int exceptionDetail0, unsigned int exceptionDetail1, unsigned int exceptionDetail2, unsigned int exceptionDetail3, unsigned int exceptionDetail4, unsigned int exceptionDetail5)
- static __forceinline__
  __device__ void optixThrowException (int exceptionCode, unsigned int exceptionDetail0, unsigned int exceptionDetail1, unsigned int exceptionDetail2, unsigned int exceptionDetail3, unsigned int exceptionDetail4, unsigned int exceptionDetail5, unsigned int exceptionDetail6)
- static __forceinline__
  __device__ void optixThrowException (int exceptionCode, unsigned int exceptionDetail0, unsigned int exceptionDetail1, unsigned int exceptionDetail2, unsigned int exceptionDetail3, unsigned int exceptionDetail4, unsigned int exceptionDetail5, unsigned int exceptionDetail6, unsigned int exceptionDetail7)
- static __forceinline__
  __device__ int optixGetExceptionCode ()
- static __forceinline__
  __device__ unsigned int optixGetExceptionDetail_0 ()
- static __forceinline__
  __device__ unsigned int optixGetExceptionDetail_1 ()

- static __forceinline__
  __device__ unsigned int optixGetExceptionDetail_2 ()
- static __forceinline__
  __device__ unsigned int optixGetExceptionDetail_3 ()
- static __forceinline__
  __device__ unsigned int optixGetExceptionDetail_4 ()
- static __forceinline__
  __device__ unsigned int optixGetExceptionDetail_5 ()
- static __forceinline__
  __device__ unsigned int optixGetExceptionDetail_6 ()
- static __forceinline__
  __device__ unsigned int optixGetExceptionDetail_7 ()
- static __forceinline__
  __device__
  OptixTraversableHandle optixGetExceptionInvalidTraversable ()
- static __forceinline__
  __device__ int optixGetExceptionInvalidSbtOffset ()
- static __forceinline__
  __device__
  OptixInvalidRayExceptionDetails optixGetExceptionInvalidRay ()
- static __forceinline__
  __device__
  OptixParameterMismatchExceptionDetails optixGetExceptionParameterMismatch ()
- static __forceinline__
  __device__ char ∗ optixGetExceptionLineInfo ()
- template<typename ReturnT , typename... ArgTypes>
  static __forceinline__
  __device__ ReturnT optixDirectCall (unsigned int sbtIndex, ArgTypes...args)
- template<typename ReturnT , typename... ArgTypes>
  static __forceinline__
  __device__ ReturnT optixContinuationCall (unsigned int sbtIndex, ArgTypes...args)
- static __forceinline__
  __device__ uint4 optixTexFootprint2D (unsigned long long tex, unsigned int texInfo, float x, float y, unsigned int ∗singleMipLevel)
- static __forceinline__
  __device__ uint4 optixTexFootprint2DLod (unsigned long long tex, unsigned int texInfo, float x, float y, float level, bool coarse, unsigned int ∗singleMipLevel)
- static __forceinline__
  __device__ uint4 optixTexFootprint2DGrad (unsigned long long tex, unsigned int texInfo, float x, float y, float dPdx_x, float dPdx_y, float dPdy_x, float dPdy_y, bool coarse, unsigned int ∗singleMipLevel)

4.1.1 Detailed Description

OptiX Device API.

4.1.2 Function Documentation

4.1.2.1 template<typename ReturnT , typename... ArgTypes> static __forceinline__ __device__ ReturnT optixContinuationCall (unsigned int *sbtIndex,* ArgTypes... *args*) `[static]`

Creates a call to the continuation callable program at the specified SBT entry.

This will call the program that was specified in the OptixProgramGroupCallables::entryFunctionNameCC in the module specified by OptixProgramGroupCallables::moduleCC. The address of the SBT entry is calculated by OptixShaderBindingTable::callablesRecordBase + (OptixShaderBindingTable::callablesRecordStrideInBytes $*$ sbtIndex). As opposed to direct callable programs, continuation callable programs are allowed to call optixTrace recursively.

Behavior is undefined if there is no continuation callable program at the specified SBT entry.

Behavior is undefined if the number of arguments that are being passed in does not match the number of parameters expected by the program that is called. In that case an exception of type OPTIX_EXCEPTION_CODE_CALLABLE_PARAMETER_MISMATCH will be thrown if OPTIX_EXCEPTION_FLAG_DEBUG was specified for the OptixPipelineCompileOptions::exceptionFlags.

Parameters

`in`	*sbtIndex*	The offset of the SBT entry of the continuation callable program to call relative to OptixShaderBindingTable::callablesRecordBase.
`in`	*args*	The arguments to pass to the continuation callable program.

4.1.2.2 template<typename ReturnT , typename... ArgTypes> static __forceinline__ __device__ ReturnT optixDirectCall (unsigned int *sbtIndex,* ArgTypes... *args*) `[static]`

Creates a call to the direct callable program at the specified SBT entry.

This will call the program that was specified in the OptixProgramGroupCallables::entryFunctionNameDC in the module specified by OptixProgramGroupCallables::moduleDC. The address of the SBT entry is calculated by OptixShaderBindingTable::callablesRecordBase + (OptixShaderBindingTable::callablesRecordStrideInBytes $*$ sbtIndex).

Behavior is undefined if there is no direct callable program at the specified SBT entry.

Behavior is undefined if the number of arguments that are being passed in does not match the number of parameters expected by the program that is called. In that case an exception of type OPTIX_EXCEPTION_CODE_CALLABLE_PARAMETER_MISMATCH will be thrown if OPTIX_EXCEPTION_FLAG_DEBUG was specified for the OptixPipelineCompileOptions::exceptionFlags.

Parameters

in	*sbtIndex*	The offset of the SBT entry of the direct callable program to call relative to OptixShaderBindingTable::callablesRecordBase.
in	*args*	The arguments to pass to the direct callable program.

4.1.2.3 static __forceinline__ __device__ unsigned int optixGetAttribute_0 ( ) [static]

Returns the attribute at slot 0.

4.1.2.4 static __forceinline__ __device__ unsigned int optixGetAttribute_1 ( ) [static]

Returns the attribute at slot 1.

4.1.2.5 static __forceinline__ __device__ unsigned int optixGetAttribute_2 ( ) [static]

Returns the attribute at slot 2.

4.1.2.6 static __forceinline__ __device__ unsigned int optixGetAttribute_3 ( ) [static]

Returns the attribute at slot 3.

4.1.2.7 static __forceinline__ __device__ unsigned int optixGetAttribute_4 ( ) [static]

Returns the attribute at slot 4.

4.1.2.8 static __forceinline__ __device__ unsigned int optixGetAttribute_5 ( ) [static]

Returns the attribute at slot 5.

4.1.2.9 static __forceinline__ __device__ unsigned int optixGetAttribute_6 ( ) [static]

Returns the attribute at slot 6.

4.1.2.10 static __forceinline__ __device__ unsigned int optixGetAttribute_7 ( ) [static]

Returns the attribute at slot 7.

4.1.2.11 static __forceinline__ __device__ void optixGetCatmullRomVertexData (OptixTraversableHandle *gas*, unsigned int *primIdx*, unsigned int *sbtGASIndex*, float *time*, float4 *data[4]*) [static]

Return the object space curve control vertex data of a CatmullRom spline curve in a Geometry Acceleration Structure (GAS) at a given motion time. To access vertex data, the GAS must be built using the flag OPTIX_BUILD_FLAG_ALLOW_RANDOM_VERTEX_ACCESS.

data[i] = {x,y,z,w} with {x,y,z} the position and w the radius of control vertex i. If motion is disabled via

OptixPipelineCompileOptions::usesMotionBlur, or the GAS does not contain motion, the time parameter is ignored.

4.1.2.12 static __forceinline__ __device__ void optixGetCubicBSplineVertexData (OptixTraversableHandle *gas*, unsigned int *primIdx*, unsigned int *sbtGASIndex*, float *time*, float4 *data[4]*) [static]

Return the object space curve control vertex data of a cubic BSpline curve in a Geometry Acceleration Structure (GAS) at a given motion time. To access vertex data, the GAS must be built using the flag OPTIX_BUILD_FLAG_ALLOW_RANDOM_VERTEX_ACCESS.

data[i] = {x,y,z,w} with {x,y,z} the position and w the radius of control vertex i. If motion is disabled via OptixPipelineCompileOptions::usesMotionBlur, or the GAS does not contain motion, the time parameter is ignored.

4.1.2.13 static __forceinline__ __device__ float optixGetCurveParameter ( ) [static]

Convenience function that returns the curve parameter.

When using OptixBuildInputCurveArray objects, during intersection the curve parameter is stored into the first attribute register.

4.1.2.14 static __forceinline__ __device__ int optixGetExceptionCode ( ) [static]

Returns the exception code.

Only available in EX.

4.1.2.15 static __forceinline__ __device__ unsigned int optixGetExceptionDetail_0 ( ) [static]

Returns the 32-bit exception detail at slot 0.

The behavior is undefined if the exception is not a user exception, or the used overload optixThrowException() did not provide the queried exception detail.

Only available in EX.

4.1.2.16 static __forceinline__ __device__ unsigned int optixGetExceptionDetail_1 ( ) [static]

Returns the 32-bit exception detail at slot 1.

See Also

optixGetExceptionDetail_0()

4.1.2.17 static __forceinline__ __device__ unsigned int optixGetExceptionDetail_2 ( ) `[static]`

Returns the 32-bit exception detail at slot 2.

See Also

optixGetExceptionDetail_0()

4.1.2.18 static __forceinline__ __device__ unsigned int optixGetExceptionDetail_3 ( ) `[static]`

Returns the 32-bit exception detail at slot 3.

See Also

optixGetExceptionDetail_0()

4.1.2.19 static __forceinline__ __device__ unsigned int optixGetExceptionDetail_4 ( ) `[static]`

Returns the 32-bit exception detail at slot 4.

See Also

optixGetExceptionDetail_0()

4.1.2.20 static __forceinline__ __device__ unsigned int optixGetExceptionDetail_5 ( ) `[static]`

Returns the 32-bit exception detail at slot 5.

See Also

optixGetExceptionDetail_0()

4.1.2.21 static __forceinline__ __device__ unsigned int optixGetExceptionDetail_6 ( ) `[static]`

Returns the 32-bit exception detail at slot 6.

See Also

optixGetExceptionDetail_0()

4.1.2.22 static __forceinline__ __device__ unsigned int optixGetExceptionDetail_7 ( ) `[static]`

Returns the 32-bit exception detail at slot 7.

See Also

optixGetExceptionDetail_0()

4.1.2.23 static __forceinline__ __device__ OptixInvalidRayExceptionDetails optixGetExceptionInvalidRay ( ) `[static]`

Returns the invalid ray for exceptions with exception code OPTIX_EXCEPTION_CODE_INVALID_RAY. Exceptions of type OPTIX_EXCEPTION_CODE_INVALID_RAY are thrown when one or more values that were passed into optixTrace are either inf or nan.

OptixInvalidRayExceptionDetails::rayTime will always be 0 if OptixPipelineCompileOptions::usesMotionBlur is 0. Values in the returned struct are all zero for all other exception codes.

Only available in EX.

4.1.2.24 static __forceinline__ __device__ int optixGetExceptionInvalidSbtOffset ( ) `[static]`

Returns the invalid sbt offset for exceptions with exception code OPTIX_EXCEPTION_CODE_TRAVERSAL_INVALID_MISS_SBT and OPTIX_EXCEPTION_CODE_TRAVERSAL_INVALID_HIT_SBT.

Returns zero for all other exception codes.

Only available in EX.

4.1.2.25 static __forceinline__ __device__ OptixTraversableHandle optixGetExceptionInvalidTraversable ( ) `[static]`

Returns the invalid traversable handle for exceptions with exception code OPTIX_EXCEPTION_CODE_TRAVERSAL_INVALID_TRAVERSABLE.

Returns zero for all other exception codes.

Only available in EX.

4.1.2.26 static __forceinline__ __device__ char* optixGetExceptionLineInfo ( ) `[static]`

Returns a string that includes information about the source location that caused the current exception.

The source location is only available for exceptions of type OPTIX_EXCEPTION_CODE_CALLABLE_PARAMETER_MISMATCH, OPTIX_EXCEPTION_CODE_UNSUPPORTED_PRIMITIVE_TYPE, OPTIX_EXCEPTION_CODE_INVALID_RAY, and for user exceptions. Line information needs to be present in the input PTX and OptixModuleCompileOptions::debugLevel may not be set to OPTIX_COMPILE_DEBUG_LEVEL_NONE.

Returns a NULL pointer if no line information is available.

Only available in EX.

4.1.2.27 static __forceinline__ __device__ OptixParameterMismatchExceptionDetails optixGetExceptionParameterMismatch ( ) `[static]`

Returns information about an exception with code OPTIX_EXCEPTION_CODE_CALLABLE_PARAMETER_MISMATCH.

Exceptions of type OPTIX_EXCEPTION_CODE_CALLABLE_PARAMETER_MISMATCH are called when the number of arguments that were passed into a call to optixDirectCall or optixContinuationCall does not match the number of parameters of the callable that is called. Note that the parameters are packed by OptiX into individual 32 bit values, so the number of expected and passed values may not correspond to the number of arguments passed into optixDirectCall or optixContinuationCall.

Values in the returned struct are all zero for all other exception codes.

Only available in EX.

4.1.2.28 static __forceinline__ __device__ unsigned int optixGetGASMotionStepCount (OptixTraversableHandle *gas*) `[static]`

Returns the number of motion steps of a GAS (see OptixMotionOptions)

4.1.2.29 static __forceinline__ __device__ float optixGetGASMotionTimeBegin (OptixTraversableHandle *gas*) `[static]`

Returns the motion begin time of a GAS (see OptixMotionOptions)

4.1.2.30 static __forceinline__ __device__ float optixGetGASMotionTimeEnd (OptixTraversableHandle *gas*) `[static]`

Returns the motion end time of a GAS (see OptixMotionOptions)

4.1.2.31 static __forceinline__ __device__ OptixTraversableHandle optixGetGASTraversableHandle ( ) `[static]`

Returns the traversable handle for the Geometry Acceleration Structure (GAS) containing the current hit. May be called from IS, AH and CH.

4.1.2.32 static __forceinline__ __device__ unsigned int optixGetHitKind ( ) `[static]`

Returns the 8 bit hit kind associated with the current hit.

Use optixGetPrimitiveType() to interpret the hit kind. For custom intersections (primitive type OPTIX_PRIMITIVE_TYPE_CUSTOM), this is the 7-bit hitKind passed to optixReportIntersection(). Hit kinds greater than 127 are reserved for built-in primitives.

Available only in AH and CH.

4.1.2.33 static __forceinline__ __device__ OptixTraversableHandle optixGetInstanceChildFromHandle (OptixTraversableHandle *handle*) `[static]`

Returns child traversable handle from an OptixInstance traversable.

Returns 0 if the traversable handle does not reference an OptixInstance.

4.1.2.34 static __forceinline__ __device__ unsigned int optixGetInstanceId ( ) [static]

Returns the OptixInstance::instanceId of the instance within the top level acceleration structure associated with the current intersection.

When building an acceleration structure using OptixBuildInputInstanceArray each OptixInstance has a user supplied instanceId. OptixInstance objects reference another acceleration structure. During traversal the acceleration structures are visited top down. In the IS and AH programs the OptixInstance::instanceId corresponding to the most recently visited OptixInstance is returned when calling optixGetInstanceId(). In CH optixGetInstanceId() returns the OptixInstance::instanceId when the hit was recorded with optixReportIntersection. In the case where there is no OptixInstance visited, optixGetInstanceId returns ∼0u

4.1.2.35 static __forceinline__ __device__ unsigned int optixGetInstanceIdFromHandle (OptixTraversableHandle *handle*) [static]

Returns instanceId from an OptixInstance traversable.

Returns 0 if the traversable handle does not reference an OptixInstance.

4.1.2.36 static __forceinline__ __device__ unsigned int optixGetInstanceIndex ( ) [static]

Returns the zero-based index of the instance within its instance acceleration structure associated with the current intersection.

In the IS and AH programs the index corresponding to the most recently visited OptixInstance is returned when calling optixGetInstanceIndex(). In CH optixGetInstanceIndex() returns the index when the hit was recorded with optixReportIntersection. In the case where there is no OptixInstance visited, optixGetInstanceIndex returns 0

4.1.2.37 static __forceinline__ __device__ const float4∗ optixGetInstanceInverseTransform-FromHandle (OptixTraversableHandle *handle*) [static]

Returns world-to-object transform from an OptixInstance traversable.

Returns 0 if the traversable handle does not reference an OptixInstance.

4.1.2.38 static __forceinline__ __device__ const float4∗ optixGetInstanceTransformFromHandle (OptixTraversableHandle *handle*) [static]

Returns object-to-world transform from an OptixInstance traversable.

Returns 0 if the traversable handle does not reference an OptixInstance.

4.1.2.39 static __forceinline__ __device__ OptixTraversableHandle optixGetInstanceTraversableFromIAS (OptixTraversableHandle *ias,*

unsigned int *instIdx*) `[static]`

Return the traversable handle of a given instance in an Instance Acceleration Structure (IAS)

4.1.2.40 static __forceinline__ __device__ uint3 optixGetLaunchDimensions ( ) `[static]`

Available in any program, it returns the dimensions of the current launch specified by optixLaunch on the host.

4.1.2.41 static __forceinline__ __device__ uint3 optixGetLaunchIndex ( ) `[static]`

Available in any program, it returns the current launch index within the launch dimensions specified by optixLaunch on the host.

The raygen program is typically only launched once per launch index.

4.1.2.42 static __forceinline__ __device__ void optixGetLinearCurveVertexData (OptixTraversableHandle *gas,* unsigned int *primIdx,* unsigned int *sbtGASIndex,* float *time,* float4 *data[2]*) `[static]`

Return the object space curve control vertex data of a linear curve in a Geometry Acceleration Structure (GAS) at a given motion time. To access vertex data, the GAS must be built using the flag OPTIX_BUILD_FLAG_ALLOW_RANDOM_VERTEX_ACCESS.

data[i] = {x,y,z,w} with {x,y,z} the position and w the radius of control vertex i. If motion is disabled via OptixPipelineCompileOptions::usesMotionBlur, or the GAS does not contain motion, the time parameter is ignored.

4.1.2.43 static __forceinline__ __device__ const OptixMatrixMotionTransform∗ optixGetMatrixMotionTransformFromHandle (OptixTraversableHandle *handle*) `[static]`

Returns a pointer to a OptixMatrixMotionTransform from its traversable handle.

Returns 0 if the traversable is not of type OPTIX_TRANSFORM_TYPE_MATRIX_MOTION_TRANSFORM.

4.1.2.44 static __forceinline__ __device__ float3 optixGetObjectRayDirection ( ) `[static]`

Returns the current object space ray direction based on the current transform stack.

Only available in IS and AH.

4.1.2.45 static __forceinline__ __device__ float3 optixGetObjectRayOrigin ( ) `[static]`

Returns the current object space ray origin based on the current transform stack.

Only available in IS and AH.

4.1.2.46 static __forceinline__ __device__ void optixGetObjectToWorldTransformMatrix (

float ***m[12]*** **)** `[static]`

Returns the object-to-world transformation matrix resulting from the current active transformation list.

The cost of this function may be proportional to the size of the transformation list.

4.1.2.47 static __forceinline__ __device__ unsigned int optixGetPayload_0 ( ) `[static]`

Reads the 32-bit payload value at slot 0.

4.1.2.48 static __forceinline__ __device__ unsigned int optixGetPayload_1 ( ) `[static]`

Reads the 32-bit payload value at slot 1.

4.1.2.49 static __forceinline__ __device__ unsigned int optixGetPayload_10 ( ) `[static]`

Reads the 32-bit payload value at slot 10.

4.1.2.50 static __forceinline__ __device__ unsigned int optixGetPayload_11 ( ) `[static]`

Reads the 32-bit payload value at slot 11.

4.1.2.51 static __forceinline__ __device__ unsigned int optixGetPayload_12 ( ) `[static]`

Reads the 32-bit payload value at slot 12.

4.1.2.52 static __forceinline__ __device__ unsigned int optixGetPayload_13 ( ) `[static]`

Reads the 32-bit payload value at slot 13.

4.1.2.53 static __forceinline__ __device__ unsigned int optixGetPayload_14 ( ) `[static]`

Reads the 32-bit payload value at slot 14.

4.1.2.54 static __forceinline__ __device__ unsigned int optixGetPayload_15 ( ) `[static]`

Reads the 32-bit payload value at slot 15.

4.1.2.55 static __forceinline__ __device__ unsigned int optixGetPayload_16 ( ) `[static]`

Reads the 32-bit payload value at slot 16.

4.1.2.56 static __forceinline__ __device__ unsigned int optixGetPayload_17 ( ) `[static]`

Reads the 32-bit payload value at slot 17.

4.1.2.57 static __forceinline__ __device__ unsigned int optixGetPayload_18 ( ) `[static]`

Reads the 32-bit payload value at slot 18.

4.1.2.58 static __forceinline__ __device__ unsigned int optixGetPayload_19 ( ) `[static]`

Reads the 32-bit payload value at slot 19.

4.1.2.59 static __forceinline__ __device__ unsigned int optixGetPayload_2 ( ) [static]

Reads the 32-bit payload value at slot 2.

4.1.2.60 static __forceinline__ __device__ unsigned int optixGetPayload_20 ( ) [static]

Reads the 32-bit payload value at slot 20.

4.1.2.61 static __forceinline__ __device__ unsigned int optixGetPayload_21 ( ) [static]

Reads the 32-bit payload value at slot 21.

4.1.2.62 static __forceinline__ __device__ unsigned int optixGetPayload_22 ( ) [static]

Reads the 32-bit payload value at slot 22.

4.1.2.63 static __forceinline__ __device__ unsigned int optixGetPayload_23 ( ) [static]

Reads the 32-bit payload value at slot 23.

4.1.2.64 static __forceinline__ __device__ unsigned int optixGetPayload_24 ( ) [static]

Reads the 32-bit payload value at slot 24.

4.1.2.65 static __forceinline__ __device__ unsigned int optixGetPayload_25 ( ) [static]

Reads the 32-bit payload value at slot 25.

4.1.2.66 static __forceinline__ __device__ unsigned int optixGetPayload_26 ( ) [static]

Reads the 32-bit payload value at slot 26.

4.1.2.67 static __forceinline__ __device__ unsigned int optixGetPayload_27 ( ) [static]

Reads the 32-bit payload value at slot 27.

4.1.2.68 static __forceinline__ __device__ unsigned int optixGetPayload_28 ( ) [static]

Reads the 32-bit payload value at slot 28.

4.1.2.69 static __forceinline__ __device__ unsigned int optixGetPayload_29 ( ) [static]

Reads the 32-bit payload value at slot 29.

4.1.2.70 static __forceinline__ __device__ unsigned int optixGetPayload_3 ( ) [static]

Reads the 32-bit payload value at slot 3.

4.1.2.71 static __forceinline__ __device__ unsigned int optixGetPayload_30 ( ) [static]

Reads the 32-bit payload value at slot 30.

4.1.2.72 static __forceinline__ __device__ unsigned int optixGetPayload_31 ( ) `[static]`

Reads the 32-bit payload value at slot 31.

4.1.2.73 static __forceinline__ __device__ unsigned int optixGetPayload_4 ( ) `[static]`

Reads the 32-bit payload value at slot 4.

4.1.2.74 static __forceinline__ __device__ unsigned int optixGetPayload_5 ( ) `[static]`

Reads the 32-bit payload value at slot 5.

4.1.2.75 static __forceinline__ __device__ unsigned int optixGetPayload_6 ( ) `[static]`

Reads the 32-bit payload value at slot 6.

4.1.2.76 static __forceinline__ __device__ unsigned int optixGetPayload_7 ( ) `[static]`

Reads the 32-bit payload value at slot 7.

4.1.2.77 static __forceinline__ __device__ unsigned int optixGetPayload_8 ( ) `[static]`

Reads the 32-bit payload value at slot 8.

4.1.2.78 static __forceinline__ __device__ unsigned int optixGetPayload_9 ( ) `[static]`

Reads the 32-bit payload value at slot 9.

4.1.2.79 static __forceinline__ __device__ unsigned int optixGetPrimitiveIndex ( ) `[static]`

For a given OptixBuildInputTriangleArray the number of primitives is defined as "(OptixBuildInputTriangleArray::indexBuffer == 0) ? OptixBuildInputTriangleArray::numVertices/3 : OptixBuildInputTriangleArray::numIndexTriplets;". For a given OptixBuildInputCustomPrimitiveArray the number of primitives is defined as numAabbs.

The primitive index returns the index into the array of primitives plus the primitiveIndexOffset.

In IS and AH this corresponds to the currently intersected primitive. In CH this corresponds to the primitive index of the closest intersected primitive.

4.1.2.80 static __forceinline__ __device__ OptixPrimitiveType optixGetPrimitiveType (unsigned int *hitKind*) `[static]`

Function interpreting the result of optixGetHitKind().

4.1.2.81 static __forceinline__ __device__ OptixPrimitiveType optixGetPrimitiveType ( ) `[static]`

Function interpreting the hit kind associated with the current optixReportIntersection.

4.1.2.82 static __forceinline__ __device__ void optixGetQuadraticBSplineVertexData (OptixTraversableHandle *gas,*

unsigned int *primIdx,*
unsigned int *sbtGASIndex,*
float *time,*
float4 *data[3]*) `[static]`

Return the object space curve control vertex data of a quadratic BSpline curve in a Geometry Acceleration Structure (GAS) at a given motion time. To access vertex data, the GAS must be built using the flag OPTIX_BUILD_FLAG_ALLOW_RANDOM_VERTEX_ACCESS.

data[i] = {x,y,z,w} with {x,y,z} the position and w the radius of control vertex i. If motion is disabled via OptixPipelineCompileOptions::usesMotionBlur, or the GAS does not contain motion, the time parameter is ignored.

4.1.2.83 static __forceinline__ __device__ unsigned int optixGetRayFlags ( ) `[static]`

Returns the rayFlags passed into optixTrace.

Only available in IS, AH, CH, MS

4.1.2.84 static __forceinline__ __device__ float optixGetRayTime ( ) `[static]`

Returns the rayTime passed into optixTrace.

Will return 0 if motion is disabled. Only available in IS, AH, CH, MS

4.1.2.85 static __forceinline__ __device__ float optixGetRayTmax ( ) `[static]`

In IS and CH returns the current smallest reported hitT or the tmax passed into optixTrace if no hit has been reported In AH returns the hitT value as passed in to optixReportIntersection In MS returns the tmax passed into optixTrace Only available in IS, AH, CH, MS.

4.1.2.86 static __forceinline__ __device__ float optixGetRayTmin ( ) `[static]`

Returns the tmin passed into optixTrace.

Only available in IS, AH, CH, MS

4.1.2.87 static __forceinline__ __device__ unsigned int optixGetRayVisibilityMask ( ) `[static]`

Returns the visibilityMask passed into optixTrace.

Only available in IS, AH, CH, MS

4.1.2.88 static __forceinline__ __device__ CUdeviceptr optixGetSbtDataPointer ( ) `[static]`

Returns the generic memory space pointer to the data region (past the header) of the currently active SBT record corresponding to the current program.

4.1.2.89 static __forceinline__ __device__ unsigned int optixGetSbtGASIndex ( ) `[static]`

Returns the Sbt GAS index of the primitive associated with the current intersection.

In IS and AH this corresponds to the currently intersected primitive. In CH this corresponds to the Sbt

GAS index of the closest intersected primitive. In EX with exception code OPTIX_EXCEPTION_CODE_TRAVERSAL_INVALID_HIT_SBT corresponds to the sbt index within the hit GAS. Returns zero for all other exceptions.

4.1.2.90 static __forceinline__ __device__ void optixGetSphereData (OptixTraversableHandle *gas,* unsigned int *primIdx,* unsigned int *sbtGASIndex,* float *time,* float4 *data[1]*) `[static]`

Return the object space sphere data, center point and radius, in a Geometry Acceleration Structure (GAS) at a given motion time. To access sphere data, the GAS must be built using the flag OPTIX_BUILD_FLAG_ALLOW_RANDOM_VERTEX_ACCESS.

data[0] = {x,y,z,w} with {x,y,z} the position of the sphere center and w the radius. If motion is disabled via OptixPipelineCompileOptions::usesMotionBlur, or the GAS does not contain motion, the time parameter is ignored.

4.1.2.91 static __forceinline__ __device__ const OptixSRTMotionTransform∗ optixGetSRTMotionTransformFromHandle (OptixTraversableHandle *handle*) `[static]`

Returns a pointer to a OptixSRTMotionTransform from its traversable handle.

Returns 0 if the traversable is not of type OPTIX_TRANSFORM_TYPE_SRT_MOTION_TRANSFORM.

4.1.2.92 static __forceinline__ __device__ const OptixStaticTransform∗ optixGetStaticTransformFromHandle (OptixTraversableHandle *handle*) `[static]`

Returns a pointer to a OptixStaticTransform from its traversable handle.

Returns 0 if the traversable is not of type OPTIX_TRANSFORM_TYPE_STATIC_TRANSFORM.

4.1.2.93 static __forceinline__ __device__ OptixTraversableHandle optixGetTransformListHandle (unsigned int *index*) `[static]`

Returns the traversable handle for a transform on the current transform list.

Only available in IS, AH, CH, EX

4.1.2.94 static __forceinline__ __device__ unsigned int optixGetTransformListSize ( ) `[static]`

Returns the number of transforms on the current transform list.

Only available in IS, AH, CH, EX

4.1.2.95 static __forceinline__ __device__ OptixTransformType optixGetTransformTypeFromHandle (OptixTraversableHandle *handle*) `[static]`

Returns the transform type of a traversable handle from a transform list.

4.1.2.96 static __forceinline__ __device__ float2 optixGetTriangleBarycentrics ( ) `[static]`

Convenience function that returns the first two attributes as floats.

When using OptixBuildInputTriangleArray objects, during intersection the barycentric coordinates are stored into the first two attribute registers.

4.1.2.97 static __forceinline__ __device__ void optixGetTriangleVertexData (OptixTraversableHandle *gas,* unsigned int *primIdx,* unsigned int *sbtGASIndex,* float *time,* float3 *data[3]*) `[static]`

Return the object space triangle vertex positions of a given triangle in a Geometry Acceleration Structure (GAS) at a given motion time. To access vertex data, the GAS must be built using the flag OPTIX_BUILD_FLAG_ALLOW_RANDOM_VERTEX_ACCESS.

If motion is disabled via OptixPipelineCompileOptions::usesMotionBlur, or the GAS does not contain motion, the time parameter is ignored.

4.1.2.98 static __forceinline__ __device__ float3 optixGetWorldRayDirection ( ) `[static]`

Returns the rayDirection passed into optixTrace.

May be more expensive to call in IS and AH than their object space counterparts, so effort should be made to use the object space ray in those programs. Only available in IS, AH, CH, MS

4.1.2.99 static __forceinline__ __device__ float3 optixGetWorldRayOrigin ( ) `[static]`

Returns the rayOrigin passed into optixTrace.

May be more expensive to call in IS and AH than their object space counterparts, so effort should be made to use the object space ray in those programs. Only available in IS, AH, CH, MS

4.1.2.100 static __forceinline__ __device__ void optixGetWorldToObjectTransformMatrix (float *m[12]*) `[static]`

Returns the world-to-object transformation matrix resulting from the current active transformation list.

The cost of this function may be proportional to the size of the transformation list.

4.1.2.101 static __forceinline__ __device__ void optixIgnoreIntersection ( ) `[static]`

Discards the hit, and returns control to the calling optixReportIntersection or built-in intersection routine.

Available only in AH.

4.1.2.102 static __forceinline__ __device__ bool optixIsBackFaceHit (unsigned int *hitKind*) [static]

Function interpreting the result of optixGetHitKind().

4.1.2.103 static __forceinline__ __device__ bool optixIsBackFaceHit ( ) [static]

Function interpreting the hit kind associated with the current optixReportIntersection.

4.1.2.104 static __forceinline__ __device__ bool optixIsFrontFaceHit (unsigned int *hitKind*) [static]

Function interpreting the result of optixGetHitKind().

4.1.2.105 static __forceinline__ __device__ bool optixIsFrontFaceHit ( ) [static]

Function interpreting the hit kind associated with the current optixReportIntersection.

4.1.2.106 static __forceinline__ __device__ bool optixIsTriangleBackFaceHit ( ) [static]

Convenience function interpreting the result of optixGetHitKind().

4.1.2.107 static __forceinline__ __device__ bool optixIsTriangleFrontFaceHit ( ) [static]

Convenience function interpreting the result of optixGetHitKind().

4.1.2.108 static __forceinline__ __device__ bool optixIsTriangleHit ( ) [static]

Convenience function interpreting the result of optixGetHitKind().

4.1.2.109 static __forceinline__ __device__ bool optixReportIntersection (float *hitT*, unsigned int *hitKind*) [static]

Reports an intersections (overload without attributes).

If optixGetRayTmin() <= hitT <= optixGetRayTmax(), the any hit program associated with this intersection program (via the SBT entry) is called. The AH program can do one of three things:

1. call optixIgnoreIntersection - no hit is recorded, optixReportIntersection returns false
2. call optixTerminateRay - hit is recorded, optixReportIntersection does not return, no further traversal occurs, and the associated closest hit program is called
3. neither - hit is recorded, optixReportIntersection returns true hitKind - Only the 7 least significant bits should be written [0..127]. Any values above 127 are reserved for built in intersection. The value can be queried with optixGetHitKind() in AH and CH.

The attributes specified with a0..a7 are available in the AH and CH programs. Note that the attributes available in the CH program correspond to the closest recorded intersection. The number of attributes in registers and memory can be configured in the pipeline.

Parameters

`in`	*hitT*	
`in`	*hitKind*	

4.1.2.110 static __forceinline__ __device__ bool optixReportIntersection (float *hitT*, unsigned int *hitKind*, unsigned int *a0*) `[static]`

Reports an intersection (overload with 1 attribute register).

See Also

optixReportIntersection(float,unsigned int)

4.1.2.111 static __forceinline__ __device__ bool optixReportIntersection (float *hitT*, unsigned int *hitKind*, unsigned int *a0*, unsigned int *a1*) `[static]`

Reports an intersection (overload with 2 attribute registers).

See Also

optixReportIntersection(float,unsigned int)

4.1.2.112 static __forceinline__ __device__ bool optixReportIntersection (float *hitT*, unsigned int *hitKind*, unsigned int *a0*, unsigned int *a1*, unsigned int *a2*) `[static]`

Reports an intersection (overload with 3 attribute registers).

See Also

optixReportIntersection(float,unsigned int)

4.1.2.113 static __forceinline__ __device__ bool optixReportIntersection (float *hitT*, unsigned int *hitKind*, unsigned int *a0*, unsigned int *a1*,

unsigned int *a2,*
unsigned int *a3*) `[static]`

Reports an intersection (overload with 4 attribute registers).

See Also

optixReportIntersection(float,unsigned int)

4.1.2.114 static __forceinline__ __device__ bool optixReportIntersection (float *hitT*, unsigned int *hitKind*, unsigned int *a0*, unsigned int *a1*, unsigned int *a2*, unsigned int *a3*, unsigned int *a4*) `[static]`

Reports an intersection (overload with 5 attribute registers).

See Also

optixReportIntersection(float,unsigned int)

4.1.2.115 static __forceinline__ __device__ bool optixReportIntersection (float *hitT*, unsigned int *hitKind*, unsigned int *a0*, unsigned int *a1*, unsigned int *a2*, unsigned int *a3*, unsigned int *a4*, unsigned int *a5*) `[static]`

Reports an intersection (overload with 6 attribute registers).

See Also

optixReportIntersection(float,unsigned int)

4.1.2.116 static __forceinline__ __device__ bool optixReportIntersection (float *hitT*, unsigned int *hitKind*, unsigned int *a0*, unsigned int *a1*, unsigned int *a2*, unsigned int *a3*,

unsigned int *a4,*
unsigned int *a5,*
unsigned int *a6*) `[static]`

Reports an intersection (overload with 7 attribute registers).

See Also

optixReportIntersection(float,unsigned int)

4.1.2.117 static __forceinline__ __device__ bool optixReportIntersection (float *hitT*, unsigned int *hitKind*, unsigned int *a0*, unsigned int *a1*, unsigned int *a2*, unsigned int *a3*, unsigned int *a4*, unsigned int *a5*, unsigned int *a6*, unsigned int *a7*) `[static]`

Reports an intersection (overload with 8 attribute registers).

See Also

optixReportIntersection(float,unsigned int)

4.1.2.118 static __forceinline__ __device__ void optixSetPayload_0 (unsigned int *p*) `[static]`

Writes the 32-bit payload value at slot 0.

4.1.2.119 static __forceinline__ __device__ void optixSetPayload_1 (unsigned int *p*) `[static]`

Writes the 32-bit payload value at slot 1.

4.1.2.120 static __forceinline__ __device__ void optixSetPayload_10 (unsigned int *p*) `[static]`

Writes the 32-bit payload value at slot 10.

4.1.2.121 static __forceinline__ __device__ void optixSetPayload_11 (unsigned int *p*) `[static]`

Writes the 32-bit payload value at slot 11.

4.1.2.122 static __forceinline__ __device__ void optixSetPayload_12 (unsigned int *p*) [static]

Writes the 32-bit payload value at slot 12.

4.1.2.123 static __forceinline__ __device__ void optixSetPayload_13 (unsigned int *p*) [static]

Writes the 32-bit payload value at slot 13.

4.1.2.124 static __forceinline__ __device__ void optixSetPayload_14 (unsigned int *p*) [static]

Writes the 32-bit payload value at slot 14.

4.1.2.125 static __forceinline__ __device__ void optixSetPayload_15 (unsigned int *p*) [static]

Writes the 32-bit payload value at slot 15.

4.1.2.126 static __forceinline__ __device__ void optixSetPayload_16 (unsigned int *p*) [static]

Writes the 32-bit payload value at slot 16.

4.1.2.127 static __forceinline__ __device__ void optixSetPayload_17 (unsigned int *p*) [static]

Writes the 32-bit payload value at slot 17.

4.1.2.128 static __forceinline__ __device__ void optixSetPayload_18 (unsigned int *p*) [static]

Writes the 32-bit payload value at slot 18.

4.1.2.129 static __forceinline__ __device__ void optixSetPayload_19 (unsigned int *p*) [static]

Writes the 32-bit payload value at slot 19.

4.1.2.130 static __forceinline__ __device__ void optixSetPayload_2 (unsigned int *p*) [static]

Writes the 32-bit payload value at slot 2.

4.1.2.131 static __forceinline__ __device__ void optixSetPayload_20 (unsigned int *p*) [static]

Writes the 32-bit payload value at slot 20.

4.1.2.132 static __forceinline__ __device__ void optixSetPayload_21 (unsigned int *p*) `[static]`

Writes the 32-bit payload value at slot 21.

4.1.2.133 static __forceinline__ __device__ void optixSetPayload_22 (unsigned int *p*) `[static]`

Writes the 32-bit payload value at slot 22.

4.1.2.134 static __forceinline__ __device__ void optixSetPayload_23 (unsigned int *p*) `[static]`

Writes the 32-bit payload value at slot 23.

4.1.2.135 static __forceinline__ __device__ void optixSetPayload_24 (unsigned int *p*) `[static]`

Writes the 32-bit payload value at slot 24.

4.1.2.136 static __forceinline__ __device__ void optixSetPayload_25 (unsigned int *p*) `[static]`

Writes the 32-bit payload value at slot 25.

4.1.2.137 static __forceinline__ __device__ void optixSetPayload_26 (unsigned int *p*) `[static]`

Writes the 32-bit payload value at slot 26.

4.1.2.138 static __forceinline__ __device__ void optixSetPayload_27 (unsigned int *p*) `[static]`

Writes the 32-bit payload value at slot 27.

4.1.2.139 static __forceinline__ __device__ void optixSetPayload_28 (unsigned int *p*) `[static]`

Writes the 32-bit payload value at slot 28.

4.1.2.140 static __forceinline__ __device__ void optixSetPayload_29 (unsigned int *p*) `[static]`

Writes the 32-bit payload value at slot 29.

4.1.2.141 static __forceinline__ __device__ void optixSetPayload_3 (unsigned int *p*) `[static]`

Writes the 32-bit payload value at slot 3.

4.1.2.142 static __forceinline__ __device__ void optixSetPayload_30 (unsigned int *p*) [static]

Writes the 32-bit payload value at slot 30.

4.1.2.143 static __forceinline__ __device__ void optixSetPayload_31 (unsigned int *p*) [static]

Writes the 32-bit payload value at slot 31.

4.1.2.144 static __forceinline__ __device__ void optixSetPayload_4 (unsigned int *p*) [static]

Writes the 32-bit payload value at slot 4.

4.1.2.145 static __forceinline__ __device__ void optixSetPayload_5 (unsigned int *p*) [static]

Writes the 32-bit payload value at slot 5.

4.1.2.146 static __forceinline__ __device__ void optixSetPayload_6 (unsigned int *p*) [static]

Writes the 32-bit payload value at slot 6.

4.1.2.147 static __forceinline__ __device__ void optixSetPayload_7 (unsigned int *p*) [static]

Writes the 32-bit payload value at slot 7.

4.1.2.148 static __forceinline__ __device__ void optixSetPayload_8 (unsigned int *p*) [static]

Writes the 32-bit payload value at slot 8.

4.1.2.149 static __forceinline__ __device__ void optixSetPayload_9 (unsigned int *p*) [static]

Writes the 32-bit payload value at slot 9.

4.1.2.150 static __forceinline__ __device__ void optixSetPayloadTypes (unsigned int *typeMask*) [static]

Specify the supported payload types for a program.

The supported types are specified as a bitwise combination of payload types. (See OptixPayloadTypeID) May only be called once per program. Must be called at the top of the program. Only available in IS, AH, CH, MS

4.1.2.151 static __forceinline__ __device__ void optixTerminateRay ( ) [static]

Record the hit, stops traversal, and proceeds to CH.

Available only in AH.

4.1.2.152 static __forceinline__ __device__ uint4 optixTexFootprint2D (unsigned long long *tex*, unsigned int *texInfo*, float *x*, float *y*, unsigned int * *singleMipLevel*) [static]

optixTexFootprint2D calculates the footprint of a corresponding 2D texture fetch (non-mipmapped).

On Turing and subsequent architectures, a texture footprint instruction allows user programs to determine the set of texels that would be accessed by an equivalent filtered texture lookup.

Parameters

in	*tex*	CUDA texture object (cast to 64-bit integer)
in	*texInfo*	Texture info packed into 32-bit integer, described below.
in	*x*	Texture coordinate
in	*y*	Texture coordinate
out	*singleMipLevel*	Result indicating whether the footprint spans only a single miplevel.

The texture info argument is a packed 32-bit integer with the following layout:

texInfo[31:29] = reserved (3 bits) texInfo[28:24] = miplevel count (5 bits) texInfo[23:20] = log2 of tile width (4 bits) texInfo[19:16] = log2 of tile height (4 bits) texInfo[15:10] = reserved (6 bits) texInfo[9:8] = horizontal wrap mode (2 bits) (CUaddress_mode) texInfo[7:6] = vertical wrap mode (2 bits) (CUaddress_mode) texInfo[5] = mipmap filter mode (1 bit) (CUfilter_mode) texInfo[4:0] = maximum anisotropy (5 bits)

Returns a 16-byte structure (as a uint4) that stores the footprint of a texture request at a particular "granularity", which has the following layout:

struct Texture2DFootprint { unsigned long long mask; unsigned int tileY : 12; unsigned int reserved1 : 4; unsigned int dx : 3; unsigned int dy : 3; unsigned int reserved2 : 2; unsigned int granularity : 4; unsigned int reserved3 : 4; unsigned int tileX : 12; unsigned int level : 4; unsigned int reserved4 : 16; };

The granularity indicates the size of texel groups that are represented by an 8x8 bitmask. For example, a granularity of 12 indicates texel groups that are 128x64 texels in size. In a footprint call, The returned granularity will either be the actual granularity of the result, or 0 if the footprint call was able to honor the requested granularity (the usual case).

level is the mip level of the returned footprint. Two footprint calls are needed to get the complete footprint when a texture call spans multiple mip levels.

mask is an 8x8 bitmask of texel groups that are covered, or partially covered, by the footprint. tileX and tileY give the starting position of the mask in 8x8 texel-group blocks. For example, suppose a granularity of 12 (128x64 texels), and tileX=3 and tileY=4. In this case, bit 0 of the mask (the low order bit) corresponds to texel group coordinates (3∗8, 4∗8), and texel coordinates (3∗8∗128, 4∗8∗64), within the specified mip level.

If nonzero, dx and dy specify a "toroidal rotation" of the bitmask. Toroidal rotation of a coordinate in the mask simply means that its value is reduced by 8. Continuing the example from above, if dx=0 and dy=0 the mask covers texel groups (3∗8, 4∗8) to (3∗8+7, 4∗8+7) inclusive. If, on the other hand, dx=2, the rightmost 2 columns in the mask have their x coordinates reduced by 8, and similarly for dy.

See the OptiX SDK for sample code that illustrates how to unpack the result.

4.1.2.153 static __forceinline__ __device__ uint4 optixTexFootprint2DGrad (unsigned long long *tex*, unsigned int *texInfo*, float *x*, float *y*, float *dPdx_x*, float *dPdx_y*, float *dPdy_x*, float *dPdy_y*, bool *coarse*, unsigned int ∗ *singleMipLevel*) `[static]`

optixTexFootprint2DGrad calculates the footprint of a corresponding 2D texture fetch (tex2DGrad)

Parameters

`in`	*tex*	CUDA texture object (cast to 64-bit integer)
`in`	*texInfo*	Texture info packed into 32-bit integer, described below.
`in`	*x*	Texture coordinate
`in`	*y*	Texture coordinate
`in`	*dPdx_x*	Derivative of x coordinte, which determines level of detail.
`in`	*dPdx_y*	Derivative of x coordinte, which determines level of detail.
`in`	*dPdy_x*	Derivative of y coordinte, which determines level of detail.
`in`	*dPdy_y*	Derivative of y coordinte, which determines level of detail.
`in`	*coarse*	Requests footprint from coarse miplevel, when the footprint spans two levels.
`out`	*singleMipLevel*	Result indicating whether the footprint spans only a single miplevel.

See Also

optixTexFootprint2D(unsigned long long,unsigned int,float,float,unsigned int∗)

4.1.2.154 static __forceinline__ __device__ uint4 optixTexFootprint2DLod (unsigned long long *tex*, unsigned int *texInfo*, float *x*,

float *y,*
float *level,*
bool *coarse,*
unsigned int * *singleMipLevel*) `[static]`

optixTexFootprint2DLod calculates the footprint of a corresponding 2D texture fetch (tex2DLod)

Parameters

`in`	*tex*	CUDA texture object (cast to 64-bit integer)
`in`	*texInfo*	Texture info packed into 32-bit integer, described below.
`in`	*x*	Texture coordinate
`in`	*y*	Texture coordinate
`in`	*level*	Level of detail (lod)
`in`	*coarse*	Requests footprint from coarse miplevel, when the footprint spans two levels.
`out`	*singleMipLevel*	Result indicating whether the footprint spans only a single miplevel.

See Also

optixTexFootprint2D(unsigned long long,unsigned int,float,float,unsigned int*)

4.1.2.155 static __forceinline__ __device__ void optixThrowException (int *exceptionCode*) `[static]`

Throws a user exception with the given exception code (overload without exception details).

The exception code must be in the range from 0 to 2^30 - 1. Up to 8 optional exception details can be passed. They can be queried in the EX program using optixGetExceptionDetail_0() to ..._8().

The exception details must not be used to encode pointers to the stack since the current stack is not preserved in the EX program.

Not available in EX.

Parameters

`in`	*exceptionCode*	The exception code to be thrown.

4.1.2.156 static __forceinline__ __device__ void optixThrowException (int *exceptionCode*, unsigned int *exceptionDetail0*) `[static]`

Throws a user exception with the given exception code (overload with 1 exception detail).

See Also

optixThrowException(int)

4.1.2.157 static __forceinline__ __device__ void optixThrowException (int *exceptionCode*, unsigned int *exceptionDetail0*, unsigned int *exceptionDetail1*) `[static]`

Throws a user exception with the given exception code (overload with 2 exception details).

See Also

optixThrowException(int)

4.1.2.158 static __forceinline__ __device__ void optixThrowException (int *exceptionCode*, unsigned int *exceptionDetail0*, unsigned int *exceptionDetail1*, unsigned int *exceptionDetail2*) `[static]`

Throws a user exception with the given exception code (overload with 3 exception details).

See Also

optixThrowException(int)

4.1.2.159 static __forceinline__ __device__ void optixThrowException (int *exceptionCode*, unsigned int *exceptionDetail0*, unsigned int *exceptionDetail1*, unsigned int *exceptionDetail2*, unsigned int *exceptionDetail3*) `[static]`

Throws a user exception with the given exception code (overload with 4 exception details).

See Also

optixThrowException(int)

4.1.2.160 static __forceinline__ __device__ void optixThrowException (int *exceptionCode*, unsigned int *exceptionDetail0*, unsigned int *exceptionDetail1*, unsigned int *exceptionDetail2*, unsigned int *exceptionDetail3*, unsigned int *exceptionDetail4*) `[static]`

Throws a user exception with the given exception code (overload with 5 exception details).

See Also

optixThrowException(int)

4.1.2.161 static __forceinline__ __device__ void optixThrowException (int *exceptionCode*, unsigned int *exceptionDetail0*, unsigned int *exceptionDetail1*, unsigned int *exceptionDetail2*, unsigned int *exceptionDetail3*, unsigned int *exceptionDetail4*, unsigned int *exceptionDetail5*) `[static]`

Throws a user exception with the given exception code (overload with 6 exception details).

See Also

optixThrowException(int)

4.1.2.162 static __forceinline__ __device__ void optixThrowException (int *exceptionCode*, unsigned int *exceptionDetail0*, unsigned int *exceptionDetail1*, unsigned int *exceptionDetail2*, unsigned int *exceptionDetail3*, unsigned int *exceptionDetail4*, unsigned int *exceptionDetail5*, unsigned int *exceptionDetail6*) `[static]`

Throws a user exception with the given exception code (overload with 7 exception details).

See Also

optixThrowException(int)

4.1.2.163 static __forceinline__ __device__ void optixThrowException (int *exceptionCode*, unsigned int *exceptionDetail0*, unsigned int *exceptionDetail1*, unsigned int *exceptionDetail2*, unsigned int *exceptionDetail3*, unsigned int *exceptionDetail4*, unsigned int *exceptionDetail5*, unsigned int *exceptionDetail6*, unsigned int *exceptionDetail7*) `[static]`

Throws a user exception with the given exception code (overload with 8 exception details).

See Also

optixThrowException(int)

4.1.2.164 template<typename... Payload> static __forceinline__ __device__ void optixTrace (OptixTraversableHandle *handle,* float3 *rayOrigin,* float3 *rayDirection,* float *tmin,* float *tmax,* float *rayTime,* OptixVisibilityMask *visibilityMask,* unsigned int *rayFlags,* unsigned int *SBToffset,* unsigned int *SBTstride,* unsigned int *missSBTIndex,* Payload &... *payload*) `[static]`

Initiates a ray tracing query starting with the given traversable.

Parameters

`in`	*handle*	
`in`	*rayOrigin*	
`in`	*rayDirection*	
`in`	*tmin*	
`in`	*tmax*	
`in`	*rayTime*	
`in`	*visibilityMask*	really only 8 bits
`in`	*rayFlags*	really only 8 bits, combination of OptixRayFlags
`in`	*SBToffset*	really only 8 bits
`in`	*SBTstride*	really only 8 bits
`in`	*missSBTIndex*	specifies the miss program invoked on a miss
`out`	*payload*	up to 32 unsigned int values that hold the payload

4.1.2.165 template<typename... Payload> static __forceinline__ __device__ void optixTrace (OptixPayloadTypeID *type,* OptixTraversableHandle *handle,* float3 *rayOrigin,* float3 *rayDirection,* float *tmin,* float *tmax,* float *rayTime,* OptixVisibilityMask *visibilityMask,* unsigned int *rayFlags,*

unsigned int *SBToffset,*
unsigned int *SBTstride,*
unsigned int *missSBTIndex,*
Payload &... *payload*) `[static]`

Initiates a ray tracing query starting with the given traversable.

Parameters

`in`	*type*	
`in`	*handle*	
`in`	*rayOrigin*	
`in`	*rayDirection*	
`in`	*tmin*	
`in`	*tmax*	
`in`	*rayTime*	
`in`	*visibilityMask*	really only 8 bits
`in`	*rayFlags*	really only 8 bits, combination of OptixRayFlags
`in`	*SBToffset*	really only 8 bits
`in`	*SBTstride*	really only 8 bits
`in`	*missSBTIndex*	specifies the miss program invoked on a miss
`out`	*payload*	up to 32 unsigned int values that hold the payload

4.1.2.166 static __forceinline__ __device__ float3 optixTransformNormalFromObjectToWorldSpace (float3 *normal*) `[static]`

Transforms the normal using object-to-world transformation matrix resulting from the current active transformation list.

The cost of this function may be proportional to the size of the transformation list.

4.1.2.167 static __forceinline__ __device__ float3 optixTransformNormalFromWorldToObjectSpace (float3 *normal*) `[static]`

Transforms the normal using world-to-object transformation matrix resulting from the current active transformation list.

The cost of this function may be proportional to the size of the transformation list.

4.1.2.168 static __forceinline__ __device__ float3 optixTransformPointFromObjectToWorldSpace (

float3 *point*) `[static]`

Transforms the point using object-to-world transformation matrix resulting from the current active transformation list.

The cost of this function may be proportional to the size of the transformation list.

4.1.2.169 static __forceinline__ __device__ float3 optixTransformPointFromWorldToObjectSpace (float3 *point*) [static]

Transforms the point using world-to-object transformation matrix resulting from the current active transformation list.

The cost of this function may be proportional to the size of the transformation list.

4.1.2.170 static __forceinline__ __device__ float3 optixTransformVectorFromObjectToWorldSpace (float3 *vec*) [static]

Transforms the vector using object-to-world transformation matrix resulting from the current active transformation list.

The cost of this function may be proportional to the size of the transformation list.

4.1.2.171 static __forceinline__ __device__ float3 optixTransformVectorFromWorldToObjectSpace (float3 *vec*) [static]

Transforms the vector using world-to-object transformation matrix resulting from the current active transformation list.

The cost of this function may be proportional to the size of the transformation list.

4.1.2.172 static __forceinline__ __device__ unsigned int optixUndefinedValue ( ) [static]

Returns an undefined value.

4.2 Host API

Modules

- Error handling
- Device context
- Pipelines
- Modules
- Tasks
- Program groups
- Launches
- Acceleration structures
- Denoiser

4.2.1 Detailed Description

OptiX Host API.

4.3 Error handling

Functions

- const char ∗ optixGetErrorName (OptixResult result)
- const char ∗ optixGetErrorString (OptixResult result)

4.3.1 Detailed Description

4.3.2 Function Documentation

4.3.2.1 const char∗ optixGetErrorName (OptixResult *result*)

Returns a string containing the name of an error code in the enum.

Output is a string representation of the enum. For example "OPTIX_SUCCESS" for OPTIX_SUCCESS and "OPTIX_ERROR_INVALID_VALUE" for OPTIX_ERROR_INVALID_VALUE.

If the error code is not recognized, "Unrecognized OptixResult code" is returned.

Parameters

`in`	*result*	OptixResult enum to generate string name for

See Also

optixGetErrorString

4.3.2.2 const char∗ optixGetErrorString (OptixResult *result*)

Returns the description string for an error code.

Output is a string description of the enum. For example "Success" for OPTIX_SUCCESS and "Invalid value" for OPTIX_ERROR_INVALID_VALUE.

If the error code is not recognized, "Unrecognized OptixResult code" is returned.

Parameters

`in`	*result*	OptixResult enum to generate string description for

See Also

optixGetErrorName

4.4 Device context

Functions

- OptixResult optixDeviceContextCreate (CUcontext fromContext, const OptixDeviceContextOptions ∗options, OptixDeviceContext ∗context)
- OptixResult optixDeviceContextDestroy (OptixDeviceContext context)
- OptixResult optixDeviceContextGetProperty (OptixDeviceContext context, OptixDeviceProperty property, void ∗value, size_t sizeInBytes)
- OptixResult optixDeviceContextSetLogCallback (OptixDeviceContext context, OptixLogCallback callbackFunction, void ∗callbackData, unsigned int callbackLevel)
- OptixResult optixDeviceContextSetCacheEnabled (OptixDeviceContext context, int enabled)
- OptixResult optixDeviceContextSetCacheLocation (OptixDeviceContext context, const char ∗location)
- OptixResult optixDeviceContextSetCacheDatabaseSizes (OptixDeviceContext context, size_t lowWaterMark, size_t highWaterMark)
- OptixResult optixDeviceContextGetCacheEnabled (OptixDeviceContext context, int ∗enabled)
- OptixResult optixDeviceContextGetCacheLocation (OptixDeviceContext context, char ∗location, size_t locationSize)
- OptixResult optixDeviceContextGetCacheDatabaseSizes (OptixDeviceContext context, size_t ∗lowWaterMark, size_t ∗highWaterMark)

4.4.1 Detailed Description

4.4.2 Function Documentation

4.4.2.1 OptixResult optixDeviceContextCreate (CUcontext *fromContext,* const OptixDeviceContextOptions ∗ *options,* OptixDeviceContext ∗ *context*)

Create a device context associated with the CUDA context specified with 'fromContext'.

If zero is specified for 'fromContext', OptiX will use the current CUDA context. The CUDA context should be initialized before calling optixDeviceContextCreate.

Parameters

in	*fromContext*	
in	*options*	
out	*context*	

Returns

- OPTIX_ERROR_CUDA_NOT_INITIALIZED If using zero for 'fromContext' and CUDA has not been initialized yet on the calling thread.
- OPTIX_ERROR_CUDA_ERROR CUDA operation failed.

- OPTIX_ERROR_HOST_OUT_OF_MEMORY Heap allocation failed.
- OPTIX_ERROR_INTERNAL_ERROR Internal error

4.4.2.2 OptixResult optixDeviceContextDestroy (OptixDeviceContext *context*)

Destroys all CPU and GPU state associated with the device.

It will attempt to block on CUDA streams that have launch work outstanding.

Any API objects, such as OptixModule and OptixPipeline, not already destroyed will be destroyed.

Thread safety: A device context must not be destroyed while it is still in use by concurrent API calls in other threads.

4.4.2.3 OptixResult optixDeviceContextGetCacheDatabaseSizes (OptixDeviceContext *context*, size_t ∗ *lowWaterMark*, size_t ∗ *highWaterMark*)

Returns the low and high water marks for disk cache garbage collection. If the cache has been disabled by setting the environment variable OPTIX_CACHE_MAXSIZE=0, this function will return 0 for the low and high water marks.

Parameters

`in`	*context*	the device context
`out`	*lowWaterMark*	the low water mark
`out`	*highWaterMark*	the high water mark

4.4.2.4 OptixResult optixDeviceContextGetCacheEnabled (OptixDeviceContext *context*, int ∗ *enabled*)

Indicates whether the disk cache is enabled or disabled.

Parameters

`in`	*context*	the device context
`out`	*enabled*	1 if enabled, 0 if disabled

4.4.2.5 OptixResult optixDeviceContextGetCacheLocation (OptixDeviceContext *context*, char ∗ *location*,

size_t *locationSize*)

Returns the location of the disk cache. If the cache has been disabled by setting the environment variable OPTIX_CACHE_MAXSIZE=0, this function will return an empy string.

Parameters

`in`	*context*	the device context
`out`	*location*	directory of disk cache, null terminated if locationSize $>$ 0
`in`	*locationSize*	locationSize

4.4.2.6 OptixResult optixDeviceContextGetProperty (OptixDeviceContext *context*, OptixDeviceProperty *property*, void $*$ *value*, size_t *sizeInBytes*)

Query properties of a device context.

Parameters

`in`	*context*	the device context to query the property for
`in`	*property*	the property to query
`out`	*value*	pointer to the returned
`in`	*sizeInBytes*	size of output

4.4.2.7 OptixResult optixDeviceContextSetCacheDatabaseSizes (OptixDeviceContext *context*, size_t *lowWaterMark*, size_t *highWaterMark*)

Sets the low and high water marks for disk cache garbage collection.

Garbage collection is triggered when a new entry is written to the cache and the current cache data size plus the size of the cache entry that is about to be inserted exceeds the high water mark. Garbage collection proceeds until the size reaches the low water mark. Garbage collection will always free enough space to insert the new entry without exceeding the low water mark. Setting either limit to zero will disable garbage collection. An error will be returned if both limits are non-zero and the high water mark is smaller than the low water mark.

Note that garbage collection is performed only on writes to the disk cache. No garbage collection is triggered on disk cache initialization or immediately when calling this function, but on subsequent inserting of data into the database.

If the size of a compiled module exceeds the value configured for the high water mark and garbage collection is enabled, the module will not be added to the cache and a warning will be added to the log.

The high water mark can be overridden with the environment variable OPTIX_CACHE_MAXSIZE. The environment variable takes precedence over the function parameters. The low water mark will be set to half the value of OPTIX_CACHE_MAXSIZE. Setting OPTIX_CACHE_MAXSIZE to 0 will disable the disk cache, but will not alter the contents of the cache. Negative and non-integer values will be ignored.

Parameters

`in`	*context*	the device context
`in`	*lowWaterMark*	the low water mark
`in`	*highWaterMark*	the high water mark

4.4.2.8 OptixResult optixDeviceContextSetCacheEnabled (OptixDeviceContext *context*, int *enabled*)

Enables or disables the disk cache.

If caching was previously disabled, enabling it will attempt to initialize the disk cache database using the currently configured cache location. An error will be returned if initialization fails.

Note that no in-memory cache is used, so no caching behavior will be observed if the disk cache is disabled.

The cache can be disabled by setting the environment variable OPTIX_CACHE_MAXSIZE=0. The environment variable takes precedence over this setting. See optixDeviceContextSetCacheDatabaseSizes for additional information.

Note that the disk cache can be disabled by the environment variable, but it cannot be enabled via the environment if it is disabled via the API.

Parameters

`in`	*context*	the device context
`in`	*enabled*	1 to enabled, 0 to disable

4.4.2.9 OptixResult optixDeviceContextSetCacheLocation (OptixDeviceContext *context*, const char ∗ *location*)

Sets the location of the disk cache.

The location is specified by a directory. This directory should not be used for other purposes and will be created if it does not exist. An error will be returned if is not possible to create the disk cache at the specified location for any reason (e.g., the path is invalid or the directory is not writable). Caching will be disabled if the disk cache cannot be initialized in the new location. If caching is disabled, no error will be returned until caching is enabled. If the disk cache is located on a network file share, behavior is undefined.

The location of the disk cache can be overridden with the environment variable OPTIX_CACHE_PATH. The environment variable takes precedence over this setting.

The default location depends on the operating system:

- Windows: LOCALAPPDATA%\NVIDIA\OptixCache
- Linux: /var/tmp/OptixCache_<username> (or /tmp/OptixCache_<username> if the first choice is not usable), the underscore and username suffix are omitted if the username cannot be obtained
- MacOS X: /Library/Application Support/NVIDIA/OptixCache

Parameters

in	*context*	the device context
in	*location*	directory of disk cache

4.4.2.10 OptixResult optixDeviceContextSetLogCallback (OptixDeviceContext *context*, OptixLogCallback *callbackFunction*, void ∗ *callbackData*, unsigned int *callbackLevel*)

Sets the current log callback method.

See OptixLogCallback for more details.

Thread safety: It is guaranteed that the callback itself (callbackFunction and callbackData) are updated atomically. It is not guaranteed that the callback itself (callbackFunction and callbackData) and the callbackLevel are updated atomically. It is unspecified when concurrent API calls using the same context start to make use of the new callback method.

Parameters

in	*context*	the device context
in	*callbackFunction*	the callback function to call
in	*callbackData*	pointer to data passed to callback function while invoking it
in	*callbackLevel*	callback level

4.5 Pipelines

Functions

- OptixResult optixPipelineCreate (OptixDeviceContext context, const OptixPipelineCompileOptions ∗pipelineCompileOptions, const OptixPipelineLinkOptions ∗pipelineLinkOptions, const OptixProgramGroup ∗programGroups, unsigned int numProgramGroups, char ∗logString, size_t ∗logStringSize, OptixPipeline ∗pipeline)
- OptixResult optixPipelineDestroy (OptixPipeline pipeline)
- OptixResult optixPipelineSetStackSize (OptixPipeline pipeline, unsigned int directCallableStackSizeFromTraversal, unsigned int directCallableStackSizeFromState, unsigned int continuationStackSize, unsigned int maxTraversableGraphDepth)

4.5.1 Detailed Description

4.5.2 Function Documentation

4.5.2.1 OptixResult optixPipelineCreate (OptixDeviceContext *context*, const OptixPipelineCompileOptions ∗ *pipelineCompileOptions*, const OptixPipelineLinkOptions ∗ *pipelineLinkOptions*, const OptixProgramGroup ∗ *programGroups*, unsigned int *numProgramGroups*, char ∗ *logString*, size_t ∗ *logStringSize*, OptixPipeline ∗ *pipeline*)

logString is an optional buffer that contains compiler feedback and errors. This information is also passed to the context logger (if enabled), however it may be difficult to correlate output to the logger to specific API invocations when using multiple threads. The output to logString will only contain feedback for this specific invocation of this API call.

logStringSize as input should be a pointer to the number of bytes backing logString. Upon return it contains the length of the log message (including the null terminator) which may be greater than the input value. In this case, the log message will be truncated to fit into logString.

If logString or logStringSize are NULL, no output is written to logString. If logStringSize points to a value that is zero, no output is written. This does not affect output to the context logger if enabled.

Parameters

`in`	*context*	
`in`	*pipelineCompileOptions*	
`in`	*pipelineLinkOptions*	
`in`	*programGroups*	array of ProgramGroup objects
`in`	*numProgramGroups*	number of ProgramGroup objects

Parameters

out	*logString*	Information will be written to this string. If logStringSize $>$ 0 logString will be null terminated.
in,out	*logStringSize*	
out	*pipeline*	

4.5.2.2 OptixResult optixPipelineDestroy (OptixPipeline *pipeline*)

Thread safety: A pipeline must not be destroyed while it is still in use by concurrent API calls in other threads.

4.5.2.3 OptixResult optixPipelineSetStackSize (OptixPipeline *pipeline,* unsigned int *directCallableStackSizeFromTraversal,* unsigned int *directCallableStackSizeFromState,* unsigned int *continuationStackSize,* unsigned int *maxTraversableGraphDepth*)

Sets the stack sizes for a pipeline.

Users are encouraged to see the programming guide and the implementations of the helper functions to understand how to construct the stack sizes based on their particular needs.

If this method is not used, an internal default implementation is used. The default implementation is correct (but not necessarily optimal) as long as the maximum depth of call trees of CC and DC programs is at most 2 and no motion transforms are used.

The maxTraversableGraphDepth responds to the maximal number of traversables visited when calling trace. Every acceleration structure and motion transform count as one level of traversal. E.g., for a simple IAS (instance acceleration structure) -> GAS (geometry acceleration structure) traversal graph, the maxTraversableGraphDepth is two. For IAS -> MT (motion transform) -> GAS, the maxTraversableGraphDepth is three. Note that it does not matter whether a IAS or GAS has motion or not, it always counts as one. Launching optix with exceptions turned on (see OPTIX_EXCEPTION_FLAG_TRACE_DEPTH) will throw an exception if the specified maxTraversableGraphDepth is too small.

Parameters

in	*pipeline*	The pipeline to configure the stack size for.
in	*directCallableStackSizeFromTraversal*	The direct stack size requirement for direct callables invoked from IS or AH.
in	*directCallableStackSizeFromState*	The direct stack size requirement for direct callables invoked from RG, MS, or CH.
in	*continuationStackSize*	The continuation stack requirement.
in	*maxTraversableGraphDepth*	The maximum depth of a traversable graph passed to trace.

4.6 Modules

Functions

- OptixResult optixModuleCreateFromPTX (OptixDeviceContext context, const OptixModuleCompileOptions ∗moduleCompileOptions, const OptixPipelineCompileOptions ∗pipelineCompileOptions, const char ∗PTX, size_t PTXsize, char ∗logString, size_t ∗logStringSize, OptixModule ∗module)
- OptixResult optixModuleCreateFromPTXWithTasks (OptixDeviceContext context, const OptixModuleCompileOptions ∗moduleCompileOptions, const OptixPipelineCompileOptions ∗pipelineCompileOptions, const char ∗PTX, size_t PTXsize, char ∗logString, size_t ∗logStringSize, OptixModule ∗module, OptixTask ∗firstTask)
- OptixResult optixModuleGetCompilationState (OptixModule module, OptixModuleCompileState ∗state)
- OptixResult optixModuleDestroy (OptixModule module)
- OptixResult optixBuiltinISModuleGet (OptixDeviceContext context, const OptixModuleCompileOptions ∗moduleCompileOptions, const OptixPipelineCompileOptions ∗pipelineCompileOptions, const OptixBuiltinISOptions ∗builtinISOptions, OptixModule ∗builtinModule)

4.6.1 Detailed Description

4.6.2 Function Documentation

4.6.2.1 OptixResult optixBuiltinISModuleGet (

OptixDeviceContext *context,*
const OptixModuleCompileOptions ∗ *moduleCompileOptions,*
const OptixPipelineCompileOptions ∗ *pipelineCompileOptions,*
const OptixBuiltinISOptions ∗ *builtinISOptions,*
OptixModule ∗ *builtinModule*)

Returns a module containing the intersection program for the built-in primitive type specified by the builtinISOptions. This module must be used as the moduleIS for the OptixProgramGroupHitgroup in any SBT record for that primitive type. (The entryFunctionNameIS should be null.)

4.6.2.2 OptixResult optixModuleCreateFromPTX (

OptixDeviceContext *context,*
const OptixModuleCompileOptions ∗ *moduleCompileOptions,*
const OptixPipelineCompileOptions ∗ *pipelineCompileOptions,*
const char ∗ *PTX,*
size_t *PTXsize,*
char ∗ *logString,*
size_t ∗ *logStringSize,*

OptixModule ∗ *module*)

logString is an optional buffer that contains compiler feedback and errors. This information is also passed to the context logger (if enabled), however it may be difficult to correlate output to the logger to specific API invocations when using multiple threads. The output to logString will only contain feedback for this specific invocation of this API call.

logStringSize as input should be a pointer to the number of bytes backing logString. Upon return it contains the length of the log message (including the null terminator) which may be greater than the input value. In this case, the log message will be truncated to fit into logString.

If logString or logStringSize are NULL, no output is written to logString. If logStringSize points to a value that is zero, no output is written. This does not affect output to the context logger if enabled.

Parameters

`in`	*context*	
`in`	*moduleCompileOptions*	
`in`	*pipelineCompileOptions*	All modules in a pipeline need to use the same values for the pipeline compile options.
`in`	*PTX*	Pointer to the PTX input string.
`in`	*PTXsize*	Parsing proceeds up to PTXsize characters, or the first NUL byte, whichever occurs first.
`out`	*logString*	Information will be written to this string. If logStringSize > 0 logString will be null terminated.
`in,out`	*logStringSize*	
`out`	*module*	

Returns

OPTIX_ERROR_INVALID_VALUE - context is 0, moduleCompileOptions is 0, pipelineCompileOptions is 0, PTX is 0, module is 0.

4.6.2.3 OptixResult optixModuleCreateFromPTXWithTasks (

OptixDeviceContext *context,*
const OptixModuleCompileOptions ∗ *moduleCompileOptions,*
const OptixPipelineCompileOptions ∗ *pipelineCompileOptions,*
const char ∗ *PTX,*
size_t *PTXsize,*
char ∗ *logString,*
size_t ∗ *logStringSize,*
OptixModule ∗ *module,*
OptixTask ∗ *firstTask*)

This function is designed to do just enough work to create the OptixTask return parameter and is expected to be fast enough run without needing parallel execution. A single thread could generate all the OptixTask objects for further processing in a work pool.

Options are similar to optixModuleCreateFromPTX(), aside from the return parameter, firstTask.

The memory used to hold the PTX should be live until all tasks are finished.

It is illegal to call optixModuleDestroy() if any OptixTask objects are currently being executed. In that case OPTIX_ERROR_ILLEGAL_DURING_TASK_EXECUTE will be returned.

If an invocation of optixTaskExecute fails, the OptixModule will be marked as OPTIX_MODULE_COMPILE_STATE_IMPENDING_FAILURE if there are outstanding tasks or OPTIX_MODULE_COMPILE_STATE_FAILURE if there are no outstanding tasks. Subsequent calls to optixTaskExecute() may execute additional work to collect compilation errors generated from the input. Currently executing tasks will not necessarily be terminated immediately but at the next opportunity. Logging will continue to be directed to the logger installed with the OptixDeviceContext. If logString is provided to optixModuleCreateFromPTXWithTasks(), it will contain all the compiler feedback from all executed tasks. The lifetime of the memory pointed to by logString should extend from calling optixModuleCreateFromPTXWithTasks() to when the compilation state is either OPTIX_MODULE_COMPILE_STATE_FAILURE or OPTIX_MODULE_COMPILE_STATE_COMPLETED. OptiX will not write to the logString outside of execution of optixModuleCreateFromPTXWithTasks() or optixTaskExecute(). If the compilation state is OPTIX_MODULE_COMPILE_STATE_IMPENDING_FAILURE and no further execution of optixTaskExecute() is performed the logString may be reclaimed by the application before calling optixModuleDestroy(). The contents of logString will contain output from currently completed tasks. All OptixTask objects associated with a given OptixModule will be cleaned up when optixModuleDestroy() is called regardless of whether the compilation was successful or not. If the compilation state is OPTIX_MODULE_COMPILE_STATE_IMPENDIND_FAILURE, any unstarted OptixTask objects do not need to be executed though there is no harm doing so.

See Also

optixModuleCreateFromPTX

4.6.2.4 OptixResult optixModuleDestroy (OptixModule *module*)

Call for OptixModule objects created with optixModuleCreateFromPTX and optixModuleDeserialize.

Modules must not be destroyed while they are still used by any program group.

Thread safety: A module must not be destroyed while it is still in use by concurrent API calls in other threads.

4.6.2.5 OptixResult optixModuleGetCompilationState (OptixModule *module,* OptixModuleCompileState ∗ *state*)

When creating a module with tasks, the current state of the module can be queried using this function.

Thread safety: Safe to call from any thread until optixModuleDestroy is called.

See Also

optixModuleCreateFromPTXWithTasks

4.7 Tasks

Functions

- OptixResult optixTaskExecute (OptixTask task, OptixTask ∗additionalTasks, unsigned int maxNumAdditionalTasks, unsigned int ∗numAdditionalTasksCreated)

4.7.1 Detailed Description

4.7.2 Function Documentation

4.7.2.1 OptixResult optixTaskExecute (OptixTask *task*, OptixTask ∗ *additionalTasks*, unsigned int *maxNumAdditionalTasks*, unsigned int ∗ *numAdditionalTasksCreated*)

Each OptixTask should be executed with optixTaskExecute(). If additional parallel work is found, new OptixTask objects will be returned in additionalTasks along with the number of additional tasks in numAdditionalTasksCreated. The parameter additionalTasks should point to a user allocated array of minimum size maxNumAdditionalTasks. OptiX can generate upto maxNumAdditionalTasks additional tasks.

Each task can be executed in parallel and in any order.

Thread safety: Safe to call from any thread until optixModuleDestroy() is called for any associated task.

See Also

optixModuleCreateFromPTXWithTasks

Parameters

`in`	*task*	the OptixTask to execute
`in`	*additionalTasks*	pointer to array of OptixTask objects to be filled in
`in`	*maxNumAdditionalTasks*	maximum number of additional OptixTask objects
`out`	*numAdditionalTasksCreated*	number of OptixTask objects created by OptiX and written into #additionalTasks

4.8 Program groups

Functions

- OptixResult optixProgramGroupGetStackSize (OptixProgramGroup programGroup, OptixStackSizes ∗stackSizes)
- OptixResult optixProgramGroupCreate (OptixDeviceContext context, const OptixProgramGroupDesc ∗programDescriptions, unsigned int numProgramGroups, const OptixProgramGroupOptions ∗options, char ∗logString, size_t ∗logStringSize, OptixProgramGroup ∗programGroups)
- OptixResult optixProgramGroupDestroy (OptixProgramGroup programGroup)

4.8.1 Detailed Description

4.8.2 Function Documentation

4.8.2.1 OptixResult optixProgramGroupCreate (OptixDeviceContext *context*, const OptixProgramGroupDesc ∗ *programDescriptions*, unsigned int *numProgramGroups*, const OptixProgramGroupOptions ∗ *options*, char ∗ *logString*, size_t ∗ *logStringSize*, OptixProgramGroup ∗ *programGroups*)

logString is an optional buffer that contains compiler feedback and errors. This information is also passed to the context logger (if enabled), however it may be difficult to correlate output to the logger to specific API invocations when using multiple threads. The output to logString will only contain feedback for this specific invocation of this API call.

logStringSize as input should be a pointer to the number of bytes backing logString. Upon return it contains the length of the log message (including the null terminator) which may be greater than the input value. In this case, the log message will be truncated to fit into logString.

If logString or logStringSize are NULL, no output is written to logString. If logStringSize points to a value that is zero, no output is written. This does not affect output to the context logger if enabled.

Creates numProgramGroups OptiXProgramGroup objects from the specified OptixProgramGroupDesc array. The size of the arrays must match.

Parameters

`in`	*context*	
`in`	*programDescriptions*	N ∗ OptixProgramGroupDesc
`in`	*numProgramGroups*	N
`in`	*options*	
`out`	*logString*	Information will be written to this string. If logStringSize $>$ 0 logString will be null terminated.

Parameters

`in,out`	*logStringSize*	
`out`	*programGroups*	

4.8.2.2 OptixResult optixProgramGroupDestroy (OptixProgramGroup *programGroup*)

Thread safety: A program group must not be destroyed while it is still in use by concurrent API calls in other threads.

4.8.2.3 OptixResult optixProgramGroupGetStackSize (OptixProgramGroup *programGroup*, OptixStackSizes ∗ *stackSizes*)

Returns the stack sizes for the given program group.

Parameters

`in`	*programGroup*	the program group
`out`	*stackSizes*	the corresponding stack sizes

4.9 Launches

Functions

- OptixResult optixLaunch (OptixPipeline pipeline, CUstream stream, CUdeviceptr pipelineParams, size_t pipelineParamsSize, const OptixShaderBindingTable ∗sbt, unsigned int width, unsigned int height, unsigned int depth)
- OptixResult optixSbtRecordPackHeader (OptixProgramGroup programGroup, void ∗sbtRecordHeaderHostPointer)

4.9.1 Detailed Description

4.9.2 Function Documentation

4.9.2.1 OptixResult optixLaunch (OptixPipeline *pipeline*, CUstream *stream*, CUdeviceptr *pipelineParams*, size_t *pipelineParamsSize*, const OptixShaderBindingTable ∗ *sbt*, unsigned int *width*, unsigned int *height*, unsigned int *depth*)

Where the magic happens.

The stream and pipeline must belong to the same device context. Multiple launches may be issues in parallel from multiple threads to different streams.

pipelineParamsSize number of bytes are copied from the device memory pointed to by pipelineParams before launch. It is an error if pipelineParamsSize is greater than the size of the variable declared in modules and identified by OptixPipelineCompileOptions::pipelineLaunchParamsVariableName. If the launch params variable was optimized out or not found in the modules linked to the pipeline then the pipelineParams and pipelineParamsSize parameters are ignored.

sbt points to the shader binding table, which defines shader groupings and their resources. See the SBT spec.

Parameters

in	*pipeline*	
in	*stream*	
in	*pipelineParams*	
in	*pipelineParamsSize*	
in	*sbt*	
in	*width*	number of elements to compute
in	*height*	number of elements to compute

Parameters

`in`	*depth*	number of elements to compute

Thread safety: In the current implementation concurrent launches to the same pipeline are not supported. Concurrent launches require separate OptixPipeline objects.

4.9.2.2 OptixResult optixSbtRecordPackHeader (OptixProgramGroup *programGroup,* void $*$ *sbtRecordHeaderHostPointer*)

Parameters

`in`	*programGroup*	the program group containing the program(s)
`out`	*sbtRecordHeaderHostPointer*	the result sbt record header

4.10 Acceleration structures

Functions

- OptixResult optixAccelComputeMemoryUsage (OptixDeviceContext context, const OptixAccelBuildOptions ∗accelOptions, const OptixBuildInput ∗buildInputs, unsigned int numBuildInputs, OptixAccelBufferSizes ∗bufferSizes)
- OptixResult optixAccelBuild (OptixDeviceContext context, CUstream stream, const OptixAccelBuildOptions ∗accelOptions, const OptixBuildInput ∗buildInputs, unsigned int numBuildInputs, CUdeviceptr tempBuffer, size_t tempBufferSizeInBytes, CUdeviceptr outputBuffer, size_t outputBufferSizeInBytes, OptixTraversableHandle ∗outputHandle, const OptixAccelEmitDesc ∗emittedProperties, unsigned int numEmittedProperties)
- OptixResult optixAccelGetRelocationInfo (OptixDeviceContext context, OptixTraversableHandle handle, OptixAccelRelocationInfo ∗info)
- OptixResult optixAccelCheckRelocationCompatibility (OptixDeviceContext context, const OptixAccelRelocationInfo ∗info, int ∗compatible)
- OptixResult optixAccelRelocate (OptixDeviceContext context, CUstream stream, const OptixAccelRelocationInfo ∗info, CUdeviceptr instanceTraversableHandles, size_t numInstanceTraversableHandles, CUdeviceptr targetAccel, size_t targetAccelSizeInBytes, OptixTraversableHandle ∗targetHandle)
- OptixResult optixAccelCompact (OptixDeviceContext context, CUstream stream, OptixTraversableHandle inputHandle, CUdeviceptr outputBuffer, size_t outputBufferSizeInBytes, OptixTraversableHandle ∗outputHandle)
- OptixResult optixConvertPointerToTraversableHandle (OptixDeviceContext onDevice, CUdeviceptr pointer, OptixTraversableType traversableType, OptixTraversableHandle ∗traversableHandle)

4.10.1 Detailed Description

4.10.2 Function Documentation

4.10.2.1 OptixResult optixAccelBuild (

OptixDeviceContext *context,*
CUstream *stream,*
const OptixAccelBuildOptions ∗ *accelOptions,*
const OptixBuildInput ∗ *buildInputs,*
unsigned int *numBuildInputs,*
CUdeviceptr *tempBuffer,*
size_t *tempBufferSizeInBytes,*
CUdeviceptr *outputBuffer,*
size_t *outputBufferSizeInBytes,*
OptixTraversableHandle ∗ *outputHandle,*
const OptixAccelEmitDesc ∗ *emittedProperties,*
unsigned int *numEmittedProperties*)

Parameters

in	*context*	
in	*stream*	
in	*accelOptions*	accel options
in	*buildInputs*	an array of OptixBuildInput objects
in	*numBuildInputs*	must be >= 1 for GAS, and == 1 for IAS
in	*tempBuffer*	must be a multiple of OPTIX_ACCEL_BUFFER_BYTE_ALIGNMENT
in	*tempBufferSizeInBytes*	
in	*outputBuffer*	must be a multiple of OPTIX_ACCEL_BUFFER_BYTE_ALIGNMENT
in	*outputBufferSizeInBytes*	
out	*outputHandle*	
in	*emittedProperties*	types of requested properties and output buffers
in	*numEmittedProperties*	number of post-build properties to populate (may be zero)

4.10.2.2 OptixResult optixAccelCheckRelocationCompatibility (OptixDeviceContext *context*, const OptixAccelRelocationInfo ∗ *info*, int ∗ *compatible*)

Checks if an acceleration structure built using another OptixDeviceContext (that was used to fill in 'info') is compatible with the OptixDeviceContext specified in the 'context' parameter.

Any device is always compatible with itself.

Parameters

in	*context*	
in	*info*	
out	*compatible*	If OPTIX_SUCCESS is returned 'compatible' will have the value of either: • 0: This context is not compatible with acceleration structure data associated with 'info'. • 1: This context is compatible.

4.10.2.3 OptixResult optixAccelCompact (OptixDeviceContext *context*, CUstream *stream*, OptixTraversableHandle *inputHandle*, CUdeviceptr *outputBuffer*, size_t *outputBufferSizeInBytes*,

OptixTraversableHandle ∗ ***outputHandle*** **)**

After building an acceleration structure, it can be copied in a compacted form to reduce memory. In order to be compacted, OPTIX_BUILD_FLAG_ALLOW_COMPACTION must be supplied in OptixAccelBuildOptions::buildFlags passed to optixAccelBuild.

'outputBuffer' is the pointer to where the compacted acceleration structure will be written. This pointer must be a multiple of OPTIX_ACCEL_BUFFER_BYTE_ALIGNMENT.

The size of the memory specified in 'outputBufferSizeInBytes' should be at least the value computed using the OPTIX_PROPERTY_TYPE_COMPACTED_SIZE that was reported during optixAccelBuild.

Parameters

`in`	*context*	
`in`	*stream*	
`in`	*inputHandle*	
`in`	*outputBuffer*	
`in`	*outputBufferSizeInBytes*	
`out`	*outputHandle*	

4.10.2.4 OptixResult optixAccelComputeMemoryUsage (OptixDeviceContext *context*, const OptixAccelBuildOptions ∗ *accelOptions*, const OptixBuildInput ∗ *buildInputs*, unsigned int *numBuildInputs*, OptixAccelBufferSizes ∗ *bufferSizes*)

Parameters

`in`	*context*	
`in`	*accelOptions*	options for the accel build
`in`	*buildInputs*	an array of OptixBuildInput objects
`in`	*numBuildInputs*	number of elements in buildInputs (must be at least 1)
`out`	*bufferSizes*	fills in buffer sizes

4.10.2.5 OptixResult optixAccelGetRelocationInfo (OptixDeviceContext *context*, OptixTraversableHandle *handle*, OptixAccelRelocationInfo ∗ *info*)

Obtain relocation information, stored in OptixAccelRelocationInfo, for a given context and acceleration structure's traversable handle.

The relocation information can be passed to optixAccelCheckRelocationCompatibility to determine if an acceleration structure, referenced by 'handle', can be relocated to a different device's memory space (see optixAccelCheckRelocationCompatibility).

When used with optixAccelRelocate, it provides data necessary for doing the relocation.

If the acceleration structure data associated with 'handle' is copied multiple times, the same OptixAccelRelocationInfo can also be used on all copies.

Parameters

in	*context*	
in	*handle*	
out	*info*	

Returns

OPTIX_ERROR_INVALID_VALUE will be returned for traversable handles that are not from acceleration structure builds.

4.10.2.6 OptixResult optixAccelRelocate (OptixDeviceContext *context,* CUstream *stream,* const OptixAccelRelocationInfo ∗ *info,* CUdeviceptr *instanceTraversableHandles,* size_t *numInstanceTraversableHandles,* CUdeviceptr *targetAccel,* size_t *targetAccelSizeInBytes,* OptixTraversableHandle ∗ *targetHandle*)

optixAccelRelocate is called to update the acceleration structure after it has been relocated. Relocation is necessary when the acceleration structure's location in device memory has changed. optixAccelRelocate does not copy the memory. This function only operates on the relocated memory who's new location is specified by 'targetAccel'. optixAccelRelocate also returns the new OptixTraversableHandle associated with 'targetAccel'. The original memory (source) is not required to be valid, only the OptixAccelRelocationInfo.

Before copying the data and calling optixAccelRelocate, optixAccelCheckRelocationCompatibility should be called to ensure the copy will be compatible with the destination device context.

The memory pointed to by 'targetAccel' should be allocated with the same size as the source acceleration. Similar to the 'outputBuffer' used in optixAccelBuild, this pointer must be a multiple of OPTIX_ACCEL_BUFFER_BYTE_ALIGNMENT.

The memory in 'targetAccel' must be allocated as long as the accel is in use.

When relocating an accel that contains instances, 'instanceTraversableHandles' and 'numInstanceTraversableHandles' should be supplied. These are the traversable handles of the instances. These can be used when also relocating the instances. No updates to the bounds are performed. Use optixAccelBuild to update the bounds. 'instanceTraversableHandles' and 'numInstanceTraversableHandles' may be zero when relocating bottom level accel (i.e. an accel with no instances).

Parameters

in	*context*	
in	*stream*	
in	*info*	
in	*instanceTraversableHandles*	
in	*numInstanceTraversableHandles*	
in	*targetAccel*	
in	*targetAccelSizeInBytes*	
out	*targetHandle*	

4.10.2.7 OptixResult optixConvertPointerToTraversableHandle (OptixDeviceContext *onDevice*, CUdeviceptr *pointer*, OptixTraversableType *traversableType*, OptixTraversableHandle ∗ *traversableHandle*)

Parameters

in	*onDevice*	
in	*pointer*	pointer to traversable allocated in OptixDeviceContext. This pointer must be a multiple of OPTIX_TRANSFORM_BYTE_ALIGNMENT
in	*traversableType*	Type of OptixTraversableHandle to create
out	*traversableHandle*	traversable handle. traversableHandle must be in host memory

4.11 Denoiser

Functions

- OptixResult optixDenoiserCreate (OptixDeviceContext context, OptixDenoiserModelKind modelKind, const OptixDenoiserOptions ∗options, OptixDenoiser ∗denoiser)
- OptixResult optixDenoiserCreateWithUserModel (OptixDeviceContext context, const void ∗userData, size_t userDataSizeInBytes, OptixDenoiser ∗denoiser)
- OptixResult optixDenoiserDestroy (OptixDenoiser denoiser)
- OptixResult optixDenoiserComputeMemoryResources (const OptixDenoiser denoiser, unsigned int outputWidth, unsigned int outputHeight, OptixDenoiserSizes ∗returnSizes)
- OptixResult optixDenoiserSetup (OptixDenoiser denoiser, CUstream stream, unsigned int inputWidth, unsigned int inputHeight, CUdeviceptr denoiserState, size_t denoiserStateSizeInBytes, CUdeviceptr scratch, size_t scratchSizeInBytes)
- OptixResult optixDenoiserInvoke (OptixDenoiser denoiser, CUstream stream, const OptixDenoiserParams ∗params, CUdeviceptr denoiserState, size_t denoiserStateSizeInBytes, const OptixDenoiserGuideLayer ∗guideLayer, const OptixDenoiserLayer ∗layers, unsigned int numLayers, unsigned int inputOffsetX, unsigned int inputOffsetY, CUdeviceptr scratch, size_t scratchSizeInBytes)
- OptixResult optixDenoiserComputeIntensity (OptixDenoiser denoiser, CUstream stream, const OptixImage2D ∗inputImage, CUdeviceptr outputIntensity, CUdeviceptr scratch, size_t scratchSizeInBytes)
- OptixResult optixDenoiserComputeAverageColor (OptixDenoiser denoiser, CUstream stream, const OptixImage2D ∗inputImage, CUdeviceptr outputAverageColor, CUdeviceptr scratch, size_t scratchSizeInBytes)

4.11.1 Detailed Description

4.11.2 Function Documentation

4.11.2.1 OptixResult optixDenoiserComputeAverageColor (OptixDenoiser *denoiser*, CUstream *stream*, const OptixImage2D ∗ *inputImage*, CUdeviceptr *outputAverageColor*, CUdeviceptr *scratch*, size_t *scratchSizeInBytes*)

Compute average logarithmic for each of the first three channels for the given image. When denoising tiles the intensity of the entire image should be computed, i.e. not per tile to get consistent results.

The size of scratch memory required can be queried with optixDenoiserComputeMemoryResources.

data type unsigned char is not supported for 'inputImage', it must be 3 or 4 component half/float.

Parameters

in	*denoiser*	

Parameters

in	*stream*	
in	*inputImage*	
out	*outputAverageColor*	three floats
in	*scratch*	
in	*scratchSizeInBytes*	

4.11.2.2 OptixResult optixDenoiserComputeIntensity (OptixDenoiser *denoiser*, CUstream *stream*, const OptixImage2D ∗ *inputImage*, CUdeviceptr *outputIntensity*, CUdeviceptr *scratch*, size_t *scratchSizeInBytes*)

Computes the logarithmic average intensity of the given image. The returned value 'outputIntensity' is multiplied with the RGB values of the input image/tile in optixDenoiserInvoke if given in the parameter OptixDenoiserParams::hdrIntensity (otherwise 'hdrIntensity' must be a null pointer). This is useful for denoising HDR images which are very dark or bright. When denoising tiles the intensity of the entire image should be computed, i.e. not per tile to get consistent results.

For each RGB pixel in the inputImage the intensity is calculated and summed if it is greater than 1e-8f: intensity = log(r ∗ 0.212586f + g ∗ 0.715170f + b ∗ 0.072200f). The function returns 0.18 / exp(sum of intensities / number of summed pixels). More details could be found in the Reinhard tonemapping paper: `http://www.cmap.polytechnique.fr/~peyre/cours/x2005signal/hdr_photographic.pdf`

The size of scratch memory required can be queried with optixDenoiserComputeMemoryResources.

data type unsigned char is not supported for 'inputImage', it must be 3 or 4 component half/float.

Parameters

in	*denoiser*	
in	*stream*	
in	*inputImage*	
out	*outputIntensity*	single float
in	*scratch*	
in	*scratchSizeInBytes*	

4.11.2.3 OptixResult optixDenoiserComputeMemoryResources (const OptixDenoiser *denoiser*, unsigned int *outputWidth*, unsigned int *outputHeight*,

OptixDenoiserSizes ∗ *returnSizes*)

Computes the GPU memory resources required to execute the denoiser.

Memory for state and scratch buffers must be allocated with the sizes in 'returnSizes' and scratch memory passed to optixDenoiserSetup, optixDenoiserInvoke, optixDenoiserComputeIntensity and optixDenoiserComputeAverageColor. For tiled denoising an overlap area ('overlapWindowSizeInPixels') must be added to each tile on all sides which increases the amount of memory needed to denoise a tile. In case of tiling use withOverlapScratchSizeInBytes for scratch memory size. If only full resolution images are denoised, withoutOverlapScratchSizeInBytes can be used which is always smaller than withOverlapScratchSizeInBytes.

'outputWidth' and 'outputHeight' is the dimension of the image to be denoised (without overlap in case tiling is being used). 'outputWidth' and 'outputHeight' must be greater than or equal to the dimensions passed to optixDenoiserSetup.

Parameters

in	*denoiser*	
in	*outputWidth*	
in	*outputHeight*	
out	*returnSizes*	

4.11.2.4 OptixResult optixDenoiserCreate (

OptixDeviceContext *context,*
OptixDenoiserModelKind *modelKind,*
const OptixDenoiserOptions ∗ *options,*
OptixDenoiser ∗ *denoiser*)

Creates a denoiser object with the given options, using built-in inference models.

'modelKind' selects the model used for inference. Inference for the built-in models can be guided (giving hints to improve image quality) with albedo and normal vector images in the guide layer (see 'optixDenoiserInvoke'). Use of these images must be enabled in 'OptixDenoiserOptions'.

Parameters

in	*context*	
in	*modelKind*	
in	*options*	
out	*denoiser*	

4.11.2.5 OptixResult optixDenoiserCreateWithUserModel (

OptixDeviceContext *context,*
const void ∗ *userData,*
size_t *userDataSizeInBytes,*

OptixDenoiser ∗ *denoiser*)

Creates a denoiser object with the given options, using a provided inference model.

’userData’ and ’userDataSizeInBytes’ provide a user model for inference. The memory passed in userData will be accessed only during the invocation of this function and can be freed after it returns. The user model must export only one weight set which determines both the model kind and the required set of guide images.

Parameters

`in`	*context*	
`in`	*userData*	
`in`	*userDataSizeInBytes*	
`out`	*denoiser*	

4.11.2.6 OptixResult optixDenoiserDestroy (OptixDenoiser *denoiser*)

Destroys the denoiser object and any associated host resources.

4.11.2.7 OptixResult optixDenoiserInvoke (
OptixDenoiser *denoiser,*
CUstream *stream,*
const OptixDenoiserParams ∗ *params,*
CUdeviceptr *denoiserState,*
size_t *denoiserStateSizeInBytes,*
const OptixDenoiserGuideLayer ∗ *guideLayer,*
const OptixDenoiserLayer ∗ *layers,*
unsigned int *numLayers,*
unsigned int *inputOffsetX,*
unsigned int *inputOffsetY,*
CUdeviceptr *scratch,*
size_t *scratchSizeInBytes*)

Invokes denoiser on a set of input data and produces at least one output image. State memory must be available during the execution of the denoiser (or until optixDenoiserSetup is called with a new state memory pointer). Scratch memory passed is used only for the duration of this function. Scratch and state memory sizes must have a size greater than or equal to the sizes as returned by optixDenoiserComputeMemoryResources.

’inputOffsetX’ and ’inputOffsetY’ are pixel offsets in the ’inputLayers’ image specifying the beginning of the image without overlap. When denoising an entire image without tiling there is no overlap and ’inputOffsetX’ and ’inputOffsetY’ must be zero. When denoising a tile which is adjacent to one of the four sides of the entire image the corresponding offsets must also be zero since there is no overlap at the side adjacent to the image border.

’guideLayer’ provides additional information to the denoiser. When providing albedo and normal vector guide images, the corresponding fields in the 'OptixDenoiserOptions' must be enabled, see optixDenoiserCreate. 'guideLayer' must not be null. If a guide image in 'OptixDenoiserOptions' is not enabled, the corresponding image in 'OptixDenoiserGuideLayer' is ignored.

If OPTIX_DENOISER_MODEL_KIND_TEMPORAL or OPTIX_DENOISER_MODEL_KIND_TEMPORAL_AOV is selected, a 2d flow image must be given in 'OptixDenoiserGuideLayer'. It describes for each pixel the flow from the previous to the current frame (a 2d vector in pixel space). The denoised beauty/AOV of the previous frame must be given in 'previousOutput'. If this image is not available in the first frame of a sequence, the noisy beauty/AOV from the first frame and zero flow vectors could be given as a substitute. For non-temporal model kinds the flow image in 'OptixDenoiserGuideLayer' is ignored. 'previousOutput' and 'output' may refer to the same buffer, i.e. 'previousOutput' is first read by this function and later overwritten with the denoised result. 'output' can be passed as 'previousOutput' to the next frame. In other model kinds (not temporal) 'previousOutput' is ignored.

The beauty layer must be given as the first entry in 'layers'. In AOV type model kinds (OPTIX_DENOISER_MODEL_KIND_AOV or in user defined models implementing kernel-prediction) additional layers for the AOV images can be given. In each layer the noisy input image is given in 'input', the denoised output is written into the 'output' image. input and output images may refer to the same buffer, with the restriction that the pixel formats must be identical for input and output when the blend mode is selected (see OptixDenoiserParams).

If OPTIX_DENOISER_MODEL_KIND_TEMPORAL or OPTIX_DENOISER_MODEL_KIND_TEMPORAL_AOV is selected, the denoised image from the previous frame must be given in 'previousOutput' in the layer. 'previousOutput' and 'output' may refer to the same buffer, i.e. 'previousOutput' is first read by this function and later overwritten with the denoised result. 'output' can be passed as 'previousOutput' to the next frame. In other model kinds (not temporal) 'previousOutput' is ignored.

If OPTIX_DENOISER_MODEL_KIND_TEMPORAL or OPTIX_DENOISER_MODEL_KIND_TEMPORAL_AOV is selected, the normal vector guide image must be given as 3d vectors in camera space. In the other models only the x and y channels are used and other channels are ignored.

Parameters

`in`	*denoiser*	
`in`	*stream*	
`in`	*params*	
`in`	*denoiserState*	
`in`	*denoiserStateSizeInBytes*	
`in`	*guideLayer*	
`in`	*layers*	
`in`	*numLayers*	
`in`	*inputOffsetX*	
`in`	*inputOffsetY*	
`in`	*scratch*	
`in`	*scratchSizeInBytes*	

4.11.2.8 OptixResult optixDenoiserSetup (OptixDenoiser *denoiser*, CUstream *stream*, unsigned int *inputWidth*, unsigned int *inputHeight*, CUdeviceptr *denoiserState*, size_t *denoiserStateSizeInBytes*, CUdeviceptr *scratch*, size_t *scratchSizeInBytes*)

Initializes the state required by the denoiser.

'inputWidth' and 'inputHeight' must include overlap on both sides of the image if tiling is being used. The overlap is returned by optixDenoiserComputeMemoryResources. For subsequent calls to optixDenoiserInvoke 'inputWidth' and 'inputHeight' are the maximum dimensions of the input layers. Dimensions of the input layers passed to optixDenoiserInvoke may be different in each invocation however they always must be smaller than 'inputWidth' and 'inputHeight' passed to optixDenoiserSetup.

Parameters

in	*denoiser*	
in	*stream*	
in	*inputWidth*	
in	*inputHeight*	
in	*denoiserState*	
in	*denoiserStateSizeInBytes*	
in	*scratch*	
in	*scratchSizeInBytes*	

4.12 Types

Classes

- struct OptixDeviceContextOptions
- struct OptixBuildInputTriangleArray
- struct OptixBuildInputCurveArray
- struct OptixBuildInputSphereArray
- struct OptixAabb
- struct OptixBuildInputCustomPrimitiveArray
- struct OptixBuildInputInstanceArray
- struct OptixBuildInput
- struct OptixInstance
- struct OptixMotionOptions
- struct OptixAccelBuildOptions
- struct OptixAccelBufferSizes
- struct OptixAccelEmitDesc
- struct OptixAccelRelocationInfo
- struct OptixStaticTransform
- struct OptixMatrixMotionTransform
- struct OptixSRTData
- struct OptixSRTMotionTransform
- struct OptixImage2D
- struct OptixDenoiserOptions
- struct OptixDenoiserGuideLayer
- struct OptixDenoiserLayer
- struct OptixDenoiserParams
- struct OptixDenoiserSizes
- struct OptixModuleCompileBoundValueEntry
- struct OptixPayloadType
- struct OptixModuleCompileOptions
- struct OptixProgramGroupSingleModule
- struct OptixProgramGroupHitgroup
- struct OptixProgramGroupCallables
- struct OptixProgramGroupDesc
- struct OptixProgramGroupOptions
- struct OptixPipelineCompileOptions
- struct OptixPipelineLinkOptions
- struct OptixShaderBindingTable
- struct OptixStackSizes
- struct OptixBuiltinISOptions

Macros

- #define OPTIX_SBT_RECORD_HEADER_SIZE ((size_t)32)
- #define OPTIX_SBT_RECORD_ALIGNMENT 16ull
- #define OPTIX_ACCEL_BUFFER_BYTE_ALIGNMENT 128ull
- #define OPTIX_INSTANCE_BYTE_ALIGNMENT 16ull
- #define OPTIX_AABB_BUFFER_BYTE_ALIGNMENT 8ull
- #define OPTIX_GEOMETRY_TRANSFORM_BYTE_ALIGNMENT 16ull
- #define OPTIX_TRANSFORM_BYTE_ALIGNMENT 64ull
- #define OPTIX_COMPILE_DEFAULT_MAX_REGISTER_COUNT 0
- #define OPTIX_COMPILE_DEFAULT_MAX_PAYLOAD_TYPE_COUNT 8
- #define OPTIX_COMPILE_DEFAULT_MAX_PAYLOAD_VALUE_COUNT 32

Typedefs

- typedef unsigned long long CUdeviceptr
- typedef struct
  OptixDeviceContext_t ∗ OptixDeviceContext
- typedef struct OptixModule_t ∗ OptixModule
- typedef struct
  OptixProgramGroup_t ∗ OptixProgramGroup
- typedef struct OptixPipeline_t ∗ OptixPipeline
- typedef struct OptixDenoiser_t ∗ OptixDenoiser
- typedef struct OptixTask_t ∗ OptixTask
- typedef unsigned long long OptixTraversableHandle
- typedef unsigned int OptixVisibilityMask
- typedef enum OptixResult OptixResult
- typedef enum OptixDeviceProperty OptixDeviceProperty
- typedef void(∗ OptixLogCallback)(unsigned int level, const char ∗tag, const char ∗message, void ∗cbdata)
- typedef enum
  OptixDeviceContextValidationMode OptixDeviceContextValidationMode
- typedef struct
  OptixDeviceContextOptions OptixDeviceContextOptions
- typedef enum OptixGeometryFlags OptixGeometryFlags
- typedef enum OptixHitKind OptixHitKind
- typedef enum OptixIndicesFormat OptixIndicesFormat
- typedef enum OptixVertexFormat OptixVertexFormat
- typedef enum OptixTransformFormat OptixTransformFormat
- typedef struct
  OptixBuildInputTriangleArray OptixBuildInputTriangleArray
- typedef enum OptixPrimitiveType OptixPrimitiveType

- typedef enum
  OptixPrimitiveTypeFlags OptixPrimitiveTypeFlags
- typedef enum OptixCurveEndcapFlags OptixCurveEndcapFlags
- typedef struct
  OptixBuildInputCurveArray OptixBuildInputCurveArray
- typedef struct
  OptixBuildInputSphereArray OptixBuildInputSphereArray
- typedef struct OptixAabb OptixAabb
- typedef struct
  OptixBuildInputCustomPrimitiveArray OptixBuildInputCustomPrimitiveArray
- typedef struct
  OptixBuildInputInstanceArray OptixBuildInputInstanceArray
- typedef enum OptixBuildInputType OptixBuildInputType
- typedef struct OptixBuildInput OptixBuildInput
- typedef enum OptixInstanceFlags OptixInstanceFlags
- typedef struct OptixInstance OptixInstance
- typedef enum OptixBuildFlags OptixBuildFlags
- typedef enum OptixBuildOperation OptixBuildOperation
- typedef enum OptixMotionFlags OptixMotionFlags
- typedef struct OptixMotionOptions OptixMotionOptions
- typedef struct
  OptixAccelBuildOptions OptixAccelBuildOptions
- typedef struct
  OptixAccelBufferSizes OptixAccelBufferSizes
- typedef enum OptixAccelPropertyType OptixAccelPropertyType
- typedef struct OptixAccelEmitDesc OptixAccelEmitDesc
- typedef struct
  OptixAccelRelocationInfo OptixAccelRelocationInfo
- typedef struct OptixStaticTransform OptixStaticTransform
- typedef struct
  OptixMatrixMotionTransform OptixMatrixMotionTransform
- typedef struct OptixSRTData OptixSRTData
- typedef struct
  OptixSRTMotionTransform OptixSRTMotionTransform
- typedef enum OptixTraversableType OptixTraversableType
- typedef enum OptixPixelFormat OptixPixelFormat
- typedef struct OptixImage2D OptixImage2D
- typedef enum OptixDenoiserModelKind OptixDenoiserModelKind
- typedef struct OptixDenoiserOptions OptixDenoiserOptions
- typedef struct
  OptixDenoiserGuideLayer OptixDenoiserGuideLayer
- typedef struct OptixDenoiserLayer OptixDenoiserLayer
- typedef enum OptixDenoiserAlphaMode OptixDenoiserAlphaMode

- typedef struct OptixDenoiserParams OptixDenoiserParams
- typedef struct OptixDenoiserSizes OptixDenoiserSizes
- typedef enum OptixRayFlags OptixRayFlags
- typedef enum OptixTransformType OptixTransformType
- typedef enum
  OptixTraversableGraphFlags OptixTraversableGraphFlags
- typedef enum
  OptixCompileOptimizationLevel OptixCompileOptimizationLevel
- typedef enum OptixCompileDebugLevel OptixCompileDebugLevel
- typedef enum
  OptixModuleCompileState OptixModuleCompileState
- typedef struct
  OptixModuleCompileBoundValueEntry OptixModuleCompileBoundValueEntry
- typedef enum OptixPayloadTypeID OptixPayloadTypeID
- typedef enum OptixPayloadSemantics OptixPayloadSemantics
- typedef struct OptixPayloadType OptixPayloadType
- typedef struct
  OptixModuleCompileOptions OptixModuleCompileOptions
- typedef enum OptixProgramGroupKind OptixProgramGroupKind
- typedef enum OptixProgramGroupFlags OptixProgramGroupFlags
- typedef struct
  OptixProgramGroupSingleModule OptixProgramGroupSingleModule
- typedef struct
  OptixProgramGroupHitgroup OptixProgramGroupHitgroup
- typedef struct
  OptixProgramGroupCallables OptixProgramGroupCallables
- typedef struct
  OptixProgramGroupDesc OptixProgramGroupDesc
- typedef struct
  OptixProgramGroupOptions OptixProgramGroupOptions
- typedef enum OptixExceptionCodes OptixExceptionCodes
- typedef enum OptixExceptionFlags OptixExceptionFlags
- typedef struct
  OptixPipelineCompileOptions OptixPipelineCompileOptions
- typedef struct
  OptixPipelineLinkOptions OptixPipelineLinkOptions
- typedef struct
  OptixShaderBindingTable OptixShaderBindingTable
- typedef struct OptixStackSizes OptixStackSizes
- typedef enum
  OptixQueryFunctionTableOptions OptixQueryFunctionTableOptions
- typedef OptixResult(OptixQueryFunctionTable_t)(int abiId, unsigned int numOptions, OptixQueryFunctionTableOptions ∗, const void ∗∗, void ∗functionTable, size_t sizeOfTable)
- typedef struct
  OptixBuiltinISOptions OptixBuiltinISOptions

Enumerations

- enum OptixResult {
  OPTIX_SUCCESS = 0,
  OPTIX_ERROR_INVALID_VALUE = 7001,
  OPTIX_ERROR_HOST_OUT_OF_MEMORY = 7002,
  OPTIX_ERROR_INVALID_OPERATION = 7003,
  OPTIX_ERROR_FILE_IO_ERROR = 7004,
  OPTIX_ERROR_INVALID_FILE_FORMAT = 7005,
  OPTIX_ERROR_DISK_CACHE_INVALID_PATH = 7010,
  OPTIX_ERROR_DISK_CACHE_PERMISSION_ERROR = 7011,
  OPTIX_ERROR_DISK_CACHE_DATABASE_ERROR = 7012,
  OPTIX_ERROR_DISK_CACHE_INVALID_DATA = 7013,
  OPTIX_ERROR_LAUNCH_FAILURE = 7050,
  OPTIX_ERROR_INVALID_DEVICE_CONTEXT = 7051,
  OPTIX_ERROR_CUDA_NOT_INITIALIZED = 7052,
  OPTIX_ERROR_VALIDATION_FAILURE = 7053,
  OPTIX_ERROR_INVALID_PTX = 7200,
  OPTIX_ERROR_INVALID_LAUNCH_PARAMETER = 7201,
  OPTIX_ERROR_INVALID_PAYLOAD_ACCESS = 7202,
  OPTIX_ERROR_INVALID_ATTRIBUTE_ACCESS = 7203,
  OPTIX_ERROR_INVALID_FUNCTION_USE = 7204,
  OPTIX_ERROR_INVALID_FUNCTION_ARGUMENTS = 7205,
  OPTIX_ERROR_PIPELINE_OUT_OF_CONSTANT_MEMORY = 7250,
  OPTIX_ERROR_PIPELINE_LINK_ERROR = 7251,
  OPTIX_ERROR_ILLEGAL_DURING_TASK_EXECUTE = 7270,
  OPTIX_ERROR_INTERNAL_COMPILER_ERROR = 7299,
  OPTIX_ERROR_DENOISER_MODEL_NOT_SET = 7300,
  OPTIX_ERROR_DENOISER_NOT_INITIALIZED = 7301,
  OPTIX_ERROR_ACCEL_NOT_COMPATIBLE = 7400,
  OPTIX_ERROR_PAYLOAD_TYPE_MISMATCH = 7500,
  OPTIX_ERROR_PAYLOAD_TYPE_RESOLUTION_FAILED = 7501,
  OPTIX_ERROR_PAYLOAD_TYPE_ID_INVALID = 7502,
  OPTIX_ERROR_NOT_SUPPORTED = 7800,
  OPTIX_ERROR_UNSUPPORTED_ABI_VERSION = 7801,
  OPTIX_ERROR_FUNCTION_TABLE_SIZE_MISMATCH = 7802,
  OPTIX_ERROR_INVALID_ENTRY_FUNCTION_OPTIONS = 7803,
  OPTIX_ERROR_LIBRARY_NOT_FOUND = 7804,
  OPTIX_ERROR_ENTRY_SYMBOL_NOT_FOUND = 7805,
  OPTIX_ERROR_LIBRARY_UNLOAD_FAILURE = 7806,
  OPTIX_ERROR_DEVICE_OUT_OF_MEMORY = 7807,
  OPTIX_ERROR_CUDA_ERROR = 7900,
  OPTIX_ERROR_INTERNAL_ERROR = 7990,
  OPTIX_ERROR_UNKNOWN = 7999 }
- enum OptixDeviceProperty {
  OPTIX_DEVICE_PROPERTY_LIMIT_MAX_TRACE_DEPTH = 0x2001,
  OPTIX_DEVICE_PROPERTY_LIMIT_MAX_TRAVERSABLE_GRAPH_DEPTH = 0x2002,
  OPTIX_DEVICE_PROPERTY_LIMIT_MAX_PRIMITIVES_PER_GAS = 0x2003,
  OPTIX_DEVICE_PROPERTY_LIMIT_MAX_INSTANCES_PER_IAS = 0x2004,

OPTIX_DEVICE_PROPERTY_RTCORE_VERSION = 0x2005,
OPTIX_DEVICE_PROPERTY_LIMIT_MAX_INSTANCE_ID = 0x2006,
OPTIX_DEVICE_PROPERTY_LIMIT_NUM_BITS_INSTANCE_VISIBILITY_MASK = 0x2007,
OPTIX_DEVICE_PROPERTY_LIMIT_MAX_SBT_RECORDS_PER_GAS = 0x2008,
OPTIX_DEVICE_PROPERTY_LIMIT_MAX_SBT_OFFSET = 0x2009 }

- enum OptixDeviceContextValidationMode {
  OPTIX_DEVICE_CONTEXT_VALIDATION_MODE_OFF = 0,
  OPTIX_DEVICE_CONTEXT_VALIDATION_MODE_ALL = 0xFFFFFFFF }
- enum OptixGeometryFlags {
  OPTIX_GEOMETRY_FLAG_NONE = 0,
  OPTIX_GEOMETRY_FLAG_DISABLE_ANYHIT = 1u << 0,
  OPTIX_GEOMETRY_FLAG_REQUIRE_SINGLE_ANYHIT_CALL = 1u << 1,
  OPTIX_GEOMETRY_FLAG_DISABLE_TRIANGLE_FACE_CULLING = 1u << 2 }
- enum OptixHitKind {
  OPTIX_HIT_KIND_TRIANGLE_FRONT_FACE = 0xFE,
  OPTIX_HIT_KIND_TRIANGLE_BACK_FACE = 0xFF }
- enum OptixIndicesFormat {
  OPTIX_INDICES_FORMAT_NONE = 0,
  OPTIX_INDICES_FORMAT_UNSIGNED_SHORT3 = 0x2102,
  OPTIX_INDICES_FORMAT_UNSIGNED_INT3 = 0x2103 }
- enum OptixVertexFormat {
  OPTIX_VERTEX_FORMAT_NONE = 0,
  OPTIX_VERTEX_FORMAT_FLOAT3 = 0x2121,
  OPTIX_VERTEX_FORMAT_FLOAT2 = 0x2122,
  OPTIX_VERTEX_FORMAT_HALF3 = 0x2123,
  OPTIX_VERTEX_FORMAT_HALF2 = 0x2124,
  OPTIX_VERTEX_FORMAT_SNORM16_3 = 0x2125,
  OPTIX_VERTEX_FORMAT_SNORM16_2 = 0x2126 }
- enum OptixTransformFormat {
  OPTIX_TRANSFORM_FORMAT_NONE = 0,
  OPTIX_TRANSFORM_FORMAT_MATRIX_FLOAT12 = 0x21E1 }
- enum OptixPrimitiveType {
  OPTIX_PRIMITIVE_TYPE_CUSTOM = 0x2500,
  OPTIX_PRIMITIVE_TYPE_ROUND_QUADRATIC_BSPLINE = 0x2501,
  OPTIX_PRIMITIVE_TYPE_ROUND_CUBIC_BSPLINE = 0x2502,
  OPTIX_PRIMITIVE_TYPE_ROUND_LINEAR = 0x2503,
  OPTIX_PRIMITIVE_TYPE_ROUND_CATMULLROM = 0x2504,
  OPTIX_PRIMITIVE_TYPE_SPHERE = 0x2506,
  OPTIX_PRIMITIVE_TYPE_TRIANGLE = 0x2531 }
- enum OptixPrimitiveTypeFlags {
  OPTIX_PRIMITIVE_TYPE_FLAGS_CUSTOM = 1 << 0,
  OPTIX_PRIMITIVE_TYPE_FLAGS_ROUND_QUADRATIC_BSPLINE = 1 << 1,
  OPTIX_PRIMITIVE_TYPE_FLAGS_ROUND_CUBIC_BSPLINE = 1 << 2,
  OPTIX_PRIMITIVE_TYPE_FLAGS_ROUND_LINEAR = 1 << 3,
  OPTIX_PRIMITIVE_TYPE_FLAGS_ROUND_CATMULLROM = 1 << 4,
  OPTIX_PRIMITIVE_TYPE_FLAGS_SPHERE = 1 << 6,
  OPTIX_PRIMITIVE_TYPE_FLAGS_TRIANGLE = 1 << 31 }

- enum OptixCurveEndcapFlags {
  OPTIX_CURVE_ENDCAP_DEFAULT = 0,
  OPTIX_CURVE_ENDCAP_ON = 1 << 0 }
- enum OptixBuildInputType {
  OPTIX_BUILD_INPUT_TYPE_TRIANGLES = 0x2141,
  OPTIX_BUILD_INPUT_TYPE_CUSTOM_PRIMITIVES = 0x2142,
  OPTIX_BUILD_INPUT_TYPE_INSTANCES = 0x2143,
  OPTIX_BUILD_INPUT_TYPE_INSTANCE_POINTERS = 0x2144,
  OPTIX_BUILD_INPUT_TYPE_CURVES = 0x2145,
  OPTIX_BUILD_INPUT_TYPE_SPHERES = 0x2146 }
- enum OptixInstanceFlags {
  OPTIX_INSTANCE_FLAG_NONE = 0,
  OPTIX_INSTANCE_FLAG_DISABLE_TRIANGLE_FACE_CULLING = 1u << 0,
  OPTIX_INSTANCE_FLAG_FLIP_TRIANGLE_FACING = 1u << 1,
  OPTIX_INSTANCE_FLAG_DISABLE_ANYHIT = 1u << 2,
  OPTIX_INSTANCE_FLAG_ENFORCE_ANYHIT = 1u << 3 }
- enum OptixBuildFlags {
  OPTIX_BUILD_FLAG_NONE = 0,
  OPTIX_BUILD_FLAG_ALLOW_UPDATE = 1u << 0,
  OPTIX_BUILD_FLAG_ALLOW_COMPACTION = 1u << 1,
  OPTIX_BUILD_FLAG_PREFER_FAST_TRACE = 1u << 2,
  OPTIX_BUILD_FLAG_PREFER_FAST_BUILD = 1u << 3,
  OPTIX_BUILD_FLAG_ALLOW_RANDOM_VERTEX_ACCESS = 1u << 4,
  OPTIX_BUILD_FLAG_ALLOW_RANDOM_INSTANCE_ACCESS = 1u << 5 }
- enum OptixBuildOperation {
  OPTIX_BUILD_OPERATION_BUILD = 0x2161,
  OPTIX_BUILD_OPERATION_UPDATE = 0x2162 }
- enum OptixMotionFlags {
  OPTIX_MOTION_FLAG_NONE = 0,
  OPTIX_MOTION_FLAG_START_VANISH = 1u << 0,
  OPTIX_MOTION_FLAG_END_VANISH = 1u << 1 }
- enum OptixAccelPropertyType {
  OPTIX_PROPERTY_TYPE_COMPACTED_SIZE = 0x2181,
  OPTIX_PROPERTY_TYPE_AABBS = 0x2182 }
- enum OptixTraversableType {
  OPTIX_TRAVERSABLE_TYPE_STATIC_TRANSFORM = 0x21C1,
  OPTIX_TRAVERSABLE_TYPE_MATRIX_MOTION_TRANSFORM = 0x21C2,
  OPTIX_TRAVERSABLE_TYPE_SRT_MOTION_TRANSFORM = 0x21C3 }
- enum OptixPixelFormat {
  OPTIX_PIXEL_FORMAT_HALF2 = 0x2207,
  OPTIX_PIXEL_FORMAT_HALF3 = 0x2201,
  OPTIX_PIXEL_FORMAT_HALF4 = 0x2202,
  OPTIX_PIXEL_FORMAT_FLOAT2 = 0x2208,
  OPTIX_PIXEL_FORMAT_FLOAT3 = 0x2203,
  OPTIX_PIXEL_FORMAT_FLOAT4 = 0x2204,
  OPTIX_PIXEL_FORMAT_UCHAR3 = 0x2205,
  OPTIX_PIXEL_FORMAT_UCHAR4 = 0x2206,

OPTIX_PIXEL_FORMAT_INTERNAL_GUIDE_LAYER = 0x2209 }

- enum OptixDenoiserModelKind {
  OPTIX_DENOISER_MODEL_KIND_LDR = 0x2322,
  OPTIX_DENOISER_MODEL_KIND_HDR = 0x2323,
  OPTIX_DENOISER_MODEL_KIND_AOV = 0x2324,
  OPTIX_DENOISER_MODEL_KIND_TEMPORAL = 0x2325,
  OPTIX_DENOISER_MODEL_KIND_TEMPORAL_AOV = 0x2326,
  OPTIX_DENOISER_MODEL_KIND_UPSCALE2X = 0x2327,
  OPTIX_DENOISER_MODEL_KIND_TEMPORAL_UPSCALE2X = 0x2328 }
- enum OptixDenoiserAlphaMode {
  OPTIX_DENOISER_ALPHA_MODE_COPY = 0,
  OPTIX_DENOISER_ALPHA_MODE_ALPHA_AS_AOV = 1,
  OPTIX_DENOISER_ALPHA_MODE_FULL_DENOISE_PASS = 2 }
- enum OptixRayFlags {
  OPTIX_RAY_FLAG_NONE = 0u,
  OPTIX_RAY_FLAG_DISABLE_ANYHIT = 1u << 0,
  OPTIX_RAY_FLAG_ENFORCE_ANYHIT = 1u << 1,
  OPTIX_RAY_FLAG_TERMINATE_ON_FIRST_HIT = 1u << 2,
  OPTIX_RAY_FLAG_DISABLE_CLOSESTHIT = 1u << 3,
  OPTIX_RAY_FLAG_CULL_BACK_FACING_TRIANGLES = 1u << 4,
  OPTIX_RAY_FLAG_CULL_FRONT_FACING_TRIANGLES = 1u << 5,
  OPTIX_RAY_FLAG_CULL_DISABLED_ANYHIT = 1u << 6,
  OPTIX_RAY_FLAG_CULL_ENFORCED_ANYHIT = 1u << 7 }
- enum OptixTransformType {
  OPTIX_TRANSFORM_TYPE_NONE = 0,
  OPTIX_TRANSFORM_TYPE_STATIC_TRANSFORM = 1,
  OPTIX_TRANSFORM_TYPE_MATRIX_MOTION_TRANSFORM = 2,
  OPTIX_TRANSFORM_TYPE_SRT_MOTION_TRANSFORM = 3,
  OPTIX_TRANSFORM_TYPE_INSTANCE = 4 }
- enum OptixTraversableGraphFlags {
  OPTIX_TRAVERSABLE_GRAPH_FLAG_ALLOW_ANY = 0,
  OPTIX_TRAVERSABLE_GRAPH_FLAG_ALLOW_SINGLE_GAS = 1u << 0,
  OPTIX_TRAVERSABLE_GRAPH_FLAG_ALLOW_SINGLE_LEVEL_INSTANCING = 1u << 1 }
- enum OptixCompileOptimizationLevel {
  OPTIX_COMPILE_OPTIMIZATION_DEFAULT = 0,
  OPTIX_COMPILE_OPTIMIZATION_LEVEL_0 = 0x2340,
  OPTIX_COMPILE_OPTIMIZATION_LEVEL_1 = 0x2341,
  OPTIX_COMPILE_OPTIMIZATION_LEVEL_2 = 0x2342,
  OPTIX_COMPILE_OPTIMIZATION_LEVEL_3 = 0x2343 }
- enum OptixCompileDebugLevel {
  OPTIX_COMPILE_DEBUG_LEVEL_DEFAULT = 0,
  OPTIX_COMPILE_DEBUG_LEVEL_NONE = 0x2350,
  OPTIX_COMPILE_DEBUG_LEVEL_MINIMAL = 0x2351,
  OPTIX_COMPILE_DEBUG_LEVEL_MODERATE = 0x2353,
  OPTIX_COMPILE_DEBUG_LEVEL_FULL = 0x2352 }
- enum OptixModuleCompileState {
  OPTIX_MODULE_COMPILE_STATE_NOT_STARTED = 0x2360,

OPTIX_MODULE_COMPILE_STATE_STARTED = 0x2361,
OPTIX_MODULE_COMPILE_STATE_IMPENDING_FAILURE = 0x2362,
OPTIX_MODULE_COMPILE_STATE_FAILED = 0x2363,
OPTIX_MODULE_COMPILE_STATE_COMPLETED = 0x2364 }

- enum OptixPayloadTypeID {
  OPTIX_PAYLOAD_TYPE_DEFAULT = 0,
  OPTIX_PAYLOAD_TYPE_ID_0 = (1 << 0u),
  OPTIX_PAYLOAD_TYPE_ID_1 = (1 << 1u),
  OPTIX_PAYLOAD_TYPE_ID_2 = (1 << 2u),
  OPTIX_PAYLOAD_TYPE_ID_3 = (1 << 3u),
  OPTIX_PAYLOAD_TYPE_ID_4 = (1 << 4u),
  OPTIX_PAYLOAD_TYPE_ID_5 = (1 << 5u),
  OPTIX_PAYLOAD_TYPE_ID_6 = (1 << 6u),
  OPTIX_PAYLOAD_TYPE_ID_7 = (1 << 7u) }
- enum OptixPayloadSemantics {
  OPTIX_PAYLOAD_SEMANTICS_TRACE_CALLER_NONE = 0,
  OPTIX_PAYLOAD_SEMANTICS_TRACE_CALLER_READ = 1u << 0,
  OPTIX_PAYLOAD_SEMANTICS_TRACE_CALLER_WRITE = 2u << 0,
  OPTIX_PAYLOAD_SEMANTICS_TRACE_CALLER_READ_WRITE = 3u << 0,
  OPTIX_PAYLOAD_SEMANTICS_CH_NONE = 0,
  OPTIX_PAYLOAD_SEMANTICS_CH_READ = 1u << 2,
  OPTIX_PAYLOAD_SEMANTICS_CH_WRITE = 2u << 2,
  OPTIX_PAYLOAD_SEMANTICS_CH_READ_WRITE = 3u << 2,
  OPTIX_PAYLOAD_SEMANTICS_MS_NONE = 0,
  OPTIX_PAYLOAD_SEMANTICS_MS_READ = 1u << 4,
  OPTIX_PAYLOAD_SEMANTICS_MS_WRITE = 2u << 4,
  OPTIX_PAYLOAD_SEMANTICS_MS_READ_WRITE = 3u << 4,
  OPTIX_PAYLOAD_SEMANTICS_AH_NONE = 0,
  OPTIX_PAYLOAD_SEMANTICS_AH_READ = 1u << 6,
  OPTIX_PAYLOAD_SEMANTICS_AH_WRITE = 2u << 6,
  OPTIX_PAYLOAD_SEMANTICS_AH_READ_WRITE = 3u << 6,
  OPTIX_PAYLOAD_SEMANTICS_IS_NONE = 0,
  OPTIX_PAYLOAD_SEMANTICS_IS_READ = 1u << 8,
  OPTIX_PAYLOAD_SEMANTICS_IS_WRITE = 2u << 8,
  OPTIX_PAYLOAD_SEMANTICS_IS_READ_WRITE = 3u << 8 }
- enum OptixProgramGroupKind {
  OPTIX_PROGRAM_GROUP_KIND_RAYGEN = 0x2421,
  OPTIX_PROGRAM_GROUP_KIND_MISS = 0x2422,
  OPTIX_PROGRAM_GROUP_KIND_EXCEPTION = 0x2423,
  OPTIX_PROGRAM_GROUP_KIND_HITGROUP = 0x2424,
  OPTIX_PROGRAM_GROUP_KIND_CALLABLES = 0x2425 }
- enum OptixProgramGroupFlags { OPTIX_PROGRAM_GROUP_FLAGS_NONE = 0 }
- enum OptixExceptionCodes {
  OPTIX_EXCEPTION_CODE_STACK_OVERFLOW = -1,
  OPTIX_EXCEPTION_CODE_TRACE_DEPTH_EXCEEDED = -2,
  OPTIX_EXCEPTION_CODE_TRAVERSAL_DEPTH_EXCEEDED = -3,
  OPTIX_EXCEPTION_CODE_TRAVERSAL_INVALID_TRAVERSABLE = -5,
  OPTIX_EXCEPTION_CODE_TRAVERSAL_INVALID_MISS_SBT = -6,

OPTIX_EXCEPTION_CODE_TRAVERSAL_INVALID_HIT_SBT = -7,
OPTIX_EXCEPTION_CODE_UNSUPPORTED_PRIMITIVE_TYPE = -8,
OPTIX_EXCEPTION_CODE_INVALID_RAY = -9,
OPTIX_EXCEPTION_CODE_CALLABLE_PARAMETER_MISMATCH = -10,
OPTIX_EXCEPTION_CODE_BUILTIN_IS_MISMATCH = -11,
OPTIX_EXCEPTION_CODE_CALLABLE_INVALID_SBT = -12,
OPTIX_EXCEPTION_CODE_CALLABLE_NO_DC_SBT_RECORD = -13,
OPTIX_EXCEPTION_CODE_CALLABLE_NO_CC_SBT_RECORD = -14,
OPTIX_EXCEPTION_CODE_UNSUPPORTED_SINGLE_LEVEL_GAS = -15,
OPTIX_EXCEPTION_CODE_INVALID_VALUE_ARGUMENT_0 = -16,
OPTIX_EXCEPTION_CODE_INVALID_VALUE_ARGUMENT_1 = -17,
OPTIX_EXCEPTION_CODE_INVALID_VALUE_ARGUMENT_2 = -18,
OPTIX_EXCEPTION_CODE_UNSUPPORTED_DATA_ACCESS = -32,
OPTIX_EXCEPTION_CODE_PAYLOAD_TYPE_MISMATCH = -33 }

- enum OptixExceptionFlags {
  OPTIX_EXCEPTION_FLAG_NONE = 0,
  OPTIX_EXCEPTION_FLAG_STACK_OVERFLOW = 1u << 0,
  OPTIX_EXCEPTION_FLAG_TRACE_DEPTH = 1u << 1,
  OPTIX_EXCEPTION_FLAG_USER = 1u << 2,
  OPTIX_EXCEPTION_FLAG_DEBUG = 1u << 3 }
- enum OptixQueryFunctionTableOptions {
  OPTIX_QUERY_FUNCTION_TABLE_OPTION_DUMMY = 0 }

4.12.1 Detailed Description

OptiX Types.

4.12.2 Macro Definition Documentation

4.12.2.1 #define OPTIX_AABB_BUFFER_BYTE_ALIGNMENT 8ull

Alignment requirement for OptixBuildInputCustomPrimitiveArray::aabbBuffers.

4.12.2.2 #define OPTIX_ACCEL_BUFFER_BYTE_ALIGNMENT 128ull

Alignment requirement for output and temporay buffers for acceleration structures.

4.12.2.3 #define OPTIX_COMPILE_DEFAULT_MAX_PAYLOAD_TYPE_COUNT 8

Maximum number of payload types allowed.

4.12.2.4 #define OPTIX_COMPILE_DEFAULT_MAX_PAYLOAD_VALUE_COUNT 32

Maximum number of payload values allowed.

4.12.2.5 #define OPTIX_COMPILE_DEFAULT_MAX_REGISTER_COUNT 0

Maximum number of registers allowed. Defaults to no explicit limit.

4.12.2.6 #define OPTIX_GEOMETRY_TRANSFORM_BYTE_ALIGNMENT 16ull

Alignment requirement for OptixBuildInputTriangleArray::preTransform.

4.12.2.7 #define OPTIX_INSTANCE_BYTE_ALIGNMENT 16ull

Alignment requirement for OptixBuildInputInstanceArray::instances.

4.12.2.8 #define OPTIX_SBT_RECORD_ALIGNMENT 16ull

Alignment requirement for device pointers in OptixShaderBindingTable.

4.12.2.9 #define OPTIX_SBT_RECORD_HEADER_SIZE ((size_t)32)

Size of the SBT record headers.

4.12.2.10 #define OPTIX_TRANSFORM_BYTE_ALIGNMENT 64ull

Alignment requirement for OptixStaticTransform, OptixMatrixMotionTransform, OptixSRTMotionTransform.

4.12.3 Typedef Documentation

4.12.3.1 typedef unsigned long long CUdeviceptr

CUDA device pointer.

4.12.3.2 typedef struct OptixAabb OptixAabb

AABB inputs.

4.12.3.3 typedef struct OptixAccelBufferSizes OptixAccelBufferSizes

Struct for querying builder allocation requirements.

Once queried the sizes should be used to allocate device memory of at least these sizes.

See Also

optixAccelComputeMemoryUsage()

4.12.3.4 typedef struct OptixAccelBuildOptions OptixAccelBuildOptions

Build options for acceleration structures.

See Also

optixAccelComputeMemoryUsage(), optixAccelBuild()

4.12.3.5 typedef struct OptixAccelEmitDesc OptixAccelEmitDesc

Specifies a type and output destination for emitted post-build properties.

See Also

optixAccelBuild()

4.12.3.6 typedef enum OptixAccelPropertyType OptixAccelPropertyType

Properties which can be emitted during acceleration structure build.

See Also

OptixAccelEmitDesc::type.

4.12.3.7 typedef struct OptixAccelRelocationInfo OptixAccelRelocationInfo

Used to store information related to relocation of acceleration structures.

See Also

optixAccelGetRelocationInfo(), optixAccelCheckRelocationCompatibility(), optixAccelRelocate()

4.12.3.8 typedef enum OptixBuildFlags OptixBuildFlags

Builder Options.

Used for OptixAccelBuildOptions::buildFlags. Can be or'ed together.

4.12.3.9 typedef struct OptixBuildInput OptixBuildInput

Build inputs.

All of them support motion and the size of the data arrays needs to match the number of motion steps

See Also

optixAccelComputeMemoryUsage(), optixAccelBuild()

4.12.3.10 typedef struct OptixBuildInputCurveArray OptixBuildInputCurveArray

Curve inputs.

A curve is a swept surface defined by a 3D spline curve and a varying width (radius). A curve (or "strand") of degree d (3=cubic, 2=quadratic, 1=linear) is represented by N $>$ d vertices and N width values, and comprises N - d segments. Each segment is defined by d+1 consecutive vertices. Each curve may have a different number of vertices.

OptiX describes the curve array as a list of curve segments. The primitive id is the segment number. It is the user's responsibility to maintain a mapping between curves and curve segments. Each index buffer entry i = indexBuffer[primid] specifies the start of a curve segment, represented by d+1 consecutive vertices in the vertex buffer, and d+1 consecutive widths in the width buffer. Width is interpolated the same way vertices are interpolated, that is, using the curve basis.

Each curves build input has only one SBT record. To create curves with different materials in the same BVH, use multiple build inputs.

See Also

OptixBuildInput::curveArray

4.12.3.11 typedef struct OptixBuildInputCustomPrimitiveArray OptixBuildInputCustomPrimitiveArray

Custom primitive inputs.

See Also

OptixBuildInput::customPrimitiveArray

4.12.3.12 typedef struct OptixBuildInputInstanceArray OptixBuildInputInstanceArray

Instance and instance pointer inputs.

See Also

OptixBuildInput::instanceArray

4.12.3.13 typedef struct OptixBuildInputSphereArray OptixBuildInputSphereArray

Sphere inputs.

A sphere is defined by a center point and a radius. Each center point is represented by a vertex in the vertex buffer. There is either a single radius for all spheres, or the radii are represented by entries in the radius buffer.

The vertex buffers and radius buffers point to a host array of device pointers, one per motion step. Host array size must match the number of motion keys as set in OptixMotionOptions (or an array of size 1 if OptixMotionOptions::numKeys is set to 0 or 1). Each per motion key device pointer must point to an array of vertices corresponding to the center points of the spheres, or an array of 1 or N radii. Format OPTIX_VERTEX_FORMAT_FLOAT3 is used for vertices, OPTIX_VERTEX_FORMAT_FLOAT for radii.

See Also

OptixBuildInput::sphereArray

4.12.3.14 typedef struct OptixBuildInputTriangleArray OptixBuildInputTriangleArray

Triangle inputs.

See Also

OptixBuildInput::triangleArray

4.12.3.15 typedef enum OptixBuildInputType OptixBuildInputType

Enum to distinguish the different build input types.

See Also

OptixBuildInput::type

4.12.3.16 typedef enum OptixBuildOperation OptixBuildOperation

Enum to specify the acceleration build operation.

Used in OptixAccelBuildOptions, which is then passed to optixAccelBuild and optixAccelComputeMemoryUsage, this enum indicates whether to do a build or an update of the acceleration structure.

Acceleration structure updates utilize the same acceleration structure, but with updated bounds. Updates are typically much faster than builds, however, large perturbations can degrade the quality of the acceleration structure.

See Also

optixAccelComputeMemoryUsage(), optixAccelBuild(), OptixAccelBuildOptions

4.12.3.17 typedef struct OptixBuiltinISOptions OptixBuiltinISOptions

Specifies the options for retrieving an intersection program for a built-in primitive type. The primitive type must not be OPTIX_PRIMITIVE_TYPE_CUSTOM.

See Also

optixBuiltinISModuleGet()

4.12.3.18 typedef enum OptixCompileDebugLevel OptixCompileDebugLevel

Debug levels.

See Also

OptixModuleCompileOptions::debugLevel

4.12.3.19 typedef enum OptixCompileOptimizationLevel OptixCompileOptimizationLevel

Optimization levels.

See Also

OptixModuleCompileOptions::optLevel

4.12.3.20 typedef enum OptixCurveEndcapFlags OptixCurveEndcapFlags

Curve end cap types, for non-linear curves.

4.12.3.21 typedef struct OptixDenoiser_t∗ OptixDenoiser

Opaque type representing a denoiser instance.

4.12.3.22 typedef enum OptixDenoiserAlphaMode OptixDenoiserAlphaMode

Various parameters used by the denoiser.

See Also

optixDenoiserInvoke()
optixDenoiserComputeIntensity()
optixDenoiserComputeAverageColor()

4.12.3.23 typedef struct OptixDenoiserGuideLayer OptixDenoiserGuideLayer

Guide layer for the denoiser.

See Also

optixDenoiserInvoke()

4.12.3.24 typedef struct OptixDenoiserLayer OptixDenoiserLayer

Input/Output layers for the denoiser.

See Also

optixDenoiserInvoke()

4.12.3.25 typedef enum OptixDenoiserModelKind OptixDenoiserModelKind

Model kind used by the denoiser.

See Also

optixDenoiserCreate

4.12.3.26 typedef struct OptixDenoiserOptions OptixDenoiserOptions

Options used by the denoiser.

See Also

optixDenoiserCreate()

4.12.3.27 typedef struct OptixDenoiserParams OptixDenoiserParams

4.12.3.28 typedef struct OptixDenoiserSizes OptixDenoiserSizes

Various sizes related to the denoiser.

See Also

optixDenoiserComputeMemoryResources()

4.12.3.29 typedef struct OptixDeviceContext_t* OptixDeviceContext

Opaque type representing a device context.

4.12.3.30 typedef struct OptixDeviceContextOptions OptixDeviceContextOptions

Parameters used for optixDeviceContextCreate()

See Also

optixDeviceContextCreate()

4.12.3.31 typedef enum OptixDeviceContextValidationMode OptixDeviceContextValidation-Mode

Validation mode settings.

When enabled, certain device code utilities will be enabled to provide as good debug and error checking facilities as possible.

See Also

optixDeviceContextCreate()

4.12.3.32 typedef enum OptixDeviceProperty OptixDeviceProperty

Parameters used for optixDeviceContextGetProperty()

See Also

optixDeviceContextGetProperty()

4.12.3.33 typedef enum OptixExceptionCodes OptixExceptionCodes

The following values are used to indicate which exception was thrown.

4.12.3.34 typedef enum OptixExceptionFlags OptixExceptionFlags

Exception flags.

See Also

OptixPipelineCompileOptions::exceptionFlags, OptixExceptionCodes

4.12.3.35 typedef enum OptixGeometryFlags OptixGeometryFlags

Flags used by OptixBuildInputTriangleArray::flags and #OptixBuildInput::flag and OptixBuildInputCustomPrimitiveArray::flags.

4.12.3.36 typedef enum OptixHitKind OptixHitKind

Legacy type: A subset of the hit kinds for built-in primitive intersections. It is preferred to use optixGetPrimitiveType(), together with optixIsFrontFaceHit() or optixIsBackFaceHit().

See Also

optixGetHitKind()

4.12.3.37 typedef struct OptixImage2D OptixImage2D

Image descriptor used by the denoiser.

See Also

optixDenoiserInvoke(), optixDenoiserComputeIntensity()

4.12.3.38 typedef enum OptixIndicesFormat OptixIndicesFormat

Format of indices used int OptixBuildInputTriangleArray::indexFormat.

4.12.3.39 typedef struct OptixInstance OptixInstance

Instances.

See Also

OptixBuildInputInstanceArray::instances

4.12.3.40 typedef enum OptixInstanceFlags OptixInstanceFlags

Flags set on the OptixInstance::flags.

These can be or'ed together to combine multiple flags.

4.12.3.41 typedef void(∗ OptixLogCallback)(unsigned int level, const char ∗tag, const char ∗message, void ∗cbdata)

Type of the callback function used for log messages.

Parameters

in	*level*	The log level indicates the severity of the message. See below for possible values.
in	*tag*	A terse message category description (e.g., 'SCENE STAT').
in	*message*	Null terminated log message (without newline at the end).
in	*cbdata*	Callback data that was provided with the callback pointer.

It is the users responsibility to ensure thread safety within this function.

The following log levels are defined.

0 disable Setting the callback level will disable all messages. The callback function will not be called in this case. 1 fatal A non-recoverable error. The context and/or OptiX itself might no longer be in a usable state. 2 error A recoverable error, e.g., when passing invalid call parameters. 3 warning Hints that OptiX might not behave exactly as requested by the user or may perform slower than expected. 4 print Status or progress messages.

Higher levels might occur.

See Also

optixDeviceContextSetLogCallback(), OptixDeviceContextOptions

4.12.3.42 typedef struct OptixMatrixMotionTransform OptixMatrixMotionTransform

Represents a matrix motion transformation.

The device address of instances of this type must be a multiple of OPTIX_TRANSFORM_BYTE_ALIGNMENT.

This struct, as defined here, handles only N=2 motion keys due to the fixed array length of its transform member. The following example shows how to create instances for an arbitrary number N of motion keys:

```
float matrixData[N][12];
... // setup matrixData

size_t transformSizeInBytes = sizeof( OptixMatrixMotionTransform ) + ( N-2 ) * 12
       * sizeof( float );
OptixMatrixMotionTransform* matrixMoptionTransform = (
      OptixMatrixMotionTransform*) malloc( transformSizeInBytes );
memset( matrixMoptionTransform, 0, transformSizeInBytes );

... // setup other members of matrixMoptionTransform
matrixMoptionTransform->motionOptions.numKeys
memcpy( matrixMoptionTransform->transform, matrixData, N * 12 * sizeof( float ) );

... // copy matrixMoptionTransform to device memory
free( matrixMoptionTransform )
```

See Also

optixConvertPointerToTraversableHandle()

4.12.3.43 typedef struct OptixModule_t* OptixModule

Opaque type representing a module.

4.12.3.44 typedef struct OptixModuleCompileBoundValueEntry OptixModuleCompileBoundValueEntry

Struct for specifying specializations for pipelineParams as specified in OptixPipelineCompileOptions::pipelineLaunchParamsVariableName.

The bound values are supposed to represent a constant value in the pipelineParams. OptiX will attempt to locate all loads from the pipelineParams and correlate them to the appropriate bound value, but there are cases where OptiX cannot safely or reliably do this. For example if the pointer to the pipelineParams is passed as an argument to a non-inline function or the offset of the load to the pipelineParams cannot be statically determined (e.g. accessed in a loop). No module should rely on the value being specialized in order to work correctly. The values in the pipelineParams specified on optixLaunch should match the bound value. If validation mode is enabled on the context, OptiX will verify that the bound values specified matches the values in pipelineParams specified to optixLaunch.

These values are compiled in to the module as constants. Once the constants are inserted into the code, an optimization pass will be run that will attempt to propagate the consants and remove unreachable code.

If caching is enabled, changes in these values will result in newly compiled modules.

The pipelineParamOffset and sizeInBytes must be within the bounds of the pipelineParams variable. OPTIX_ERROR_INVALID_VALUE will be returned from optixModuleCreateFromPTX otherwise.

If more than one bound value overlaps or the size of a bound value is equal to 0, an OPTIX_ERROR_INVALID_VALUE will be returned from optixModuleCreateFromPTX.

The same set of bound values do not need to be used for all modules in a pipeline, but overlapping values between modules must have the same value. OPTIX_ERROR_INVALID_VALUE will be returned from optixPipelineCreate otherwise.

See Also

OptixModuleCompileOptions

4.12.3.45 typedef struct OptixModuleCompileOptions OptixModuleCompileOptions

Compilation options for module.

See Also

optixModuleCreateFromPTX()

4.12.3.46 typedef enum OptixModuleCompileState OptixModuleCompileState

Module compilation state.

See Also

optixModuleGetCompilationState(), optixModuleCreateFromPTXWithTasks()

4.12.3.47 typedef enum OptixMotionFlags OptixMotionFlags

Enum to specify motion flags.

See Also

OptixMotionOptions::flags.

4.12.3.48 typedef struct OptixMotionOptions OptixMotionOptions

Motion options.

See Also

OptixAccelBuildOptions::motionOptions, OptixMatrixMotionTransform::motionOptions, OptixSRTMotionTransform::motionOptions

4.12.3.49 typedef enum OptixPayloadSemantics OptixPayloadSemantics

Semantic flags for a single payload word.

Used to specify the semantics of a payload word per shader type. "read": Shader of this type may read the payload word. "write": Shader of this type may write the payload word.

"trace_caller_write": Shaders may consume the value of the payload word passed to optixTrace by the caller. "trace_caller_read": The caller to optixTrace may read the payload word after the call to optixTrace.

Semantics can be bitwise combined. Combining "read" and "write" is equivalent to specifying "read_write". A payload needs to be writable by the caller or at least one shader type. A payload needs to be readable by the caller or at least one shader type after a being writable.

4.12.3.50 typedef struct OptixPayloadType OptixPayloadType

Specifies a single payload type.

4.12.3.51 typedef enum OptixPayloadTypeID OptixPayloadTypeID

Payload type identifiers.

4.12.3.52 typedef struct OptixPipeline_t* OptixPipeline

Opaque type representing a pipeline.

4.12.3.53 typedef struct OptixPipelineCompileOptions OptixPipelineCompileOptions

Compilation options for all modules of a pipeline.

Similar to OptixModuleCompileOptions, but these options here need to be equal for all modules of a pipeline.

See Also

optixModuleCreateFromPTX(), optixPipelineCreate()

4.12.3.54 typedef struct OptixPipelineLinkOptions OptixPipelineLinkOptions

Link options for a pipeline.

See Also

optixPipelineCreate()

4.12.3.55 typedef enum OptixPixelFormat OptixPixelFormat

Pixel formats used by the denoiser.

See Also

OptixImage2D::format

4.12.3.56 typedef enum OptixPrimitiveType OptixPrimitiveType

Builtin primitive types.

4.12.3.57 typedef enum OptixPrimitiveTypeFlags OptixPrimitiveTypeFlags

Builtin flags may be bitwise combined.

See Also

OptixPipelineCompileOptions::usesPrimitiveTypeFlags

4.12.3.58 typedef struct OptixProgramGroup_t* OptixProgramGroup

Opaque type representing a program group.

4.12.3.59 typedef struct OptixProgramGroupCallables OptixProgramGroupCallables

Program group representing callables.

Module and entry function name need to be valid for at least one of the two callables.

See Also

#OptixProgramGroupDesc::callables

4.12.3.60 typedef struct OptixProgramGroupDesc OptixProgramGroupDesc

Descriptor for program groups.

4.12.3.61 typedef enum OptixProgramGroupFlags OptixProgramGroupFlags

Flags for program groups.

4.12.3.62 typedef struct OptixProgramGroupHitgroup OptixProgramGroupHitgroup

Program group representing the hitgroup.

For each of the three program types, module and entry function name might both be `nullptr`.

See Also

OptixProgramGroupDesc::hitgroup

4.12.3.63 typedef enum OptixProgramGroupKind OptixProgramGroupKind

Distinguishes different kinds of program groups.

4.12.3.64 typedef struct OptixProgramGroupOptions OptixProgramGroupOptions

Program group options.

See Also

optixProgramGroupCreate()

4.12.3.65 typedef struct OptixProgramGroupSingleModule OptixProgramGroupSingleModule

Program group representing a single module.

Used for raygen, miss, and exception programs. In case of raygen and exception programs, module and entry function name need to be valid. For miss programs, module and entry function name might both be `nullptr`.

See Also

OptixProgramGroupDesc::raygen, OptixProgramGroupDesc::miss, OptixProgramGroupDesc::exception

4.12.3.66 typedef OptixResult(OptixQueryFunctionTable_t)(int abiId, unsigned int numOptions, OptixQueryFunctionTableOptions $*$, const void $**$, void $*$functionTable, size_t sizeOfTable)

Type of the function `optixQueryFunctionTable()`

4.12.3.67 typedef enum OptixQueryFunctionTableOptions OptixQueryFunctionTableOptions

Options that can be passed to `optixQueryFunctionTable()`

4.12.3.68 typedef enum OptixRayFlags OptixRayFlags

Ray flags passed to the device function optixTrace(). These affect the behavior of traversal per invocation.

See Also

optixTrace()

4.12.3.69 typedef enum OptixResult OptixResult

Result codes returned from API functions.

All host side API functions return OptixResult with the exception of optixGetErrorName and optixGetErrorString. When successful OPTIX_SUCCESS is returned. All return codes except for OPTIX_SUCCESS should be assumed to be errors as opposed to a warning.

See Also

optixGetErrorName(), optixGetErrorString()

4.12.3.70 typedef struct OptixShaderBindingTable OptixShaderBindingTable

Describes the shader binding table (SBT)

See Also

optixLaunch()

4.12.3.71 typedef struct OptixSRTData OptixSRTData

Represents an SRT transformation.

An SRT transformation can represent a smooth rotation with fewer motion keys than a matrix transformation. Each motion key is constructed from elements taken from a matrix S, a quaternion R, and a translation T.

The scaling matrix $S = \begin{bmatrix} sx & a & b & pvx \\ 0 & sy & c & pvy \\ 0 & 0 & sz & pvz \end{bmatrix}$ defines an affine transformation that can include scale, shear, and a translation. The translation allows to define the pivot point for the subsequent rotation.

The quaternion R = [qx, qy, qz, qw] describes a rotation with angular component qw = cos(theta/2) and other components [qx, qy, qz] = sin(theta/2) $*$ [ax, ay, az] where the axis [ax, ay, az] is normalized.

The translation matrix $T = \begin{bmatrix} 1 & 0 & 0 & tx \\ 0 & 1 & 0 & ty \\ 0 & 0 & 1 & tz \end{bmatrix}$ defines another translation that is applied after the rotation. Typically, this translation includes the inverse translation from the matrix S to reverse the translation for the pivot point for R.

To obtain the effective transformation at time t, the elements of the components of S, R, and T will be interpolated linearly. The components are then multiplied to obtain the combined transformation C = T $*$ R $*$ S. The transformation C is the effective object-to-world transformations at time t, and C^(-1) is the effective world-to-object transformation at time t.

See Also

OptixSRTMotionTransform::srtData, optixConvertPointerToTraversableHandle()

4.12.3.72 typedef struct OptixSRTMotionTransform OptixSRTMotionTransform

Represents an SRT motion transformation.

The device address of instances of this type must be a multiple of OPTIX_TRANSFORM_BYTE_ALIGNMENT.

This struct, as defined here, handles only N=2 motion keys due to the fixed array length of its srtData member. The following example shows how to create instances for an arbitrary number N of motion keys:

```
OptixSRTData srtData[N];
... // setup srtData
```

```
size_t transformSizeInBytes = sizeof( OptixSRTMotionTransform ) + ( N-2 ) * sizeof(
      OptixSRTData );
OptixSRTMotionTransform* srtMotionTransform = (
      OptixSRTMotionTransform*) malloc( transformSizeInBytes );
memset( srtMotionTransform, 0, transformSizeInBytes );

... // setup other members of srtMotionTransform
srtMotionTransform->motionOptions.numKeys   = N;
memcpy( srtMotionTransform->srtData, srtData, N * sizeof( OptixSRTData ) );

... // copy srtMotionTransform to device memory
free( srtMotionTransform )
```

See Also

optixConvertPointerToTraversableHandle()

4.12.3.73 typedef struct OptixStackSizes OptixStackSizes

Describes the stack size requirements of a program group.

See Also

optixProgramGroupGetStackSize()

4.12.3.74 typedef struct OptixStaticTransform OptixStaticTransform

Static transform.

The device address of instances of this type must be a multiple of OPTIX_TRANSFORM_BYTE_ALIGNMENT.

See Also

optixConvertPointerToTraversableHandle()

4.12.3.75 typedef struct OptixTask_t* OptixTask

Opaque type representing a work task.

4.12.3.76 typedef enum OptixTransformFormat OptixTransformFormat

Format of transform used in OptixBuildInputTriangleArray::transformFormat.

4.12.3.77 typedef enum OptixTransformType OptixTransformType

Transform.

OptixTransformType is used by the device function optixGetTransformTypeFromHandle() to determine the type of the OptixTraversableHandle returned from optixGetTransformListHandle().

4.12.3.78 typedef enum OptixTraversableGraphFlags OptixTraversableGraphFlags

Specifies the set of valid traversable graphs that may be passed to invocation of optixTrace(). Flags may be bitwise combined.

4.12.3.79 typedef unsigned long long OptixTraversableHandle

Traversable handle.

4.12.3.80 typedef enum OptixTraversableType OptixTraversableType

Traversable Handles.

See Also

optixConvertPointerToTraversableHandle()

4.12.3.81 typedef enum OptixVertexFormat OptixVertexFormat

Format of vertices used in OptixBuildInputTriangleArray::vertexFormat.

4.12.3.82 typedef unsigned int OptixVisibilityMask

Visibility mask.

4.12.4 Enumeration Type Documentation

4.12.4.1 enum OptixAccelPropertyType

Properties which can be emitted during acceleration structure build.

See Also

OptixAccelEmitDesc::type.

Enumerator

OPTIX_PROPERTY_TYPE_COMPACTED_SIZE Size of a compacted acceleration structure. The device pointer points to a uint64.

OPTIX_PROPERTY_TYPE_AABBS OptixAabb $*$ numMotionSteps.

4.12.4.2 enum OptixBuildFlags

Builder Options.

Used for OptixAccelBuildOptions::buildFlags. Can be or'ed together.

Enumerator

OPTIX_BUILD_FLAG_NONE No special flags set.

OPTIX_BUILD_FLAG_ALLOW_UPDATE Allow updating the build with new vertex positions with subsequent calls to optixAccelBuild.

OPTIX_BUILD_FLAG_ALLOW_COMPACTION

OPTIX_BUILD_FLAG_PREFER_FAST_TRACE

OPTIX_BUILD_FLAG_PREFER_FAST_BUILD

OPTIX_BUILD_FLAG_ALLOW_RANDOM_VERTEX_ACCESS Allow random access to build input vertices See optixGetTriangleVertexData optixGetLinearCurveVertexData optixGetQuadraticBSplineVertexData optixGetCubicBSplineVertexData optixGetCatmullRomVertexData optixGetSphereData.

OPTIX_BUILD_FLAG_ALLOW_RANDOM_INSTANCE_ACCESS Allow random access to instances See optixGetInstanceTraversableFromIAS.

4.12.4.3 enum OptixBuildInputType

Enum to distinguish the different build input types.

See Also

OptixBuildInput::type

Enumerator

OPTIX_BUILD_INPUT_TYPE_TRIANGLES Triangle inputs.
See Also

OptixBuildInputTriangleArray

OPTIX_BUILD_INPUT_TYPE_CUSTOM_PRIMITIVES Custom primitive inputs.
See Also

OptixBuildInputCustomPrimitiveArray

OPTIX_BUILD_INPUT_TYPE_INSTANCES Instance inputs.
See Also

OptixBuildInputInstanceArray

OPTIX_BUILD_INPUT_TYPE_INSTANCE_POINTERS Instance pointer inputs.
See Also

OptixBuildInputInstanceArray

OPTIX_BUILD_INPUT_TYPE_CURVES Curve inputs.
See Also

OptixBuildInputCurveArray

OPTIX_BUILD_INPUT_TYPE_SPHERES Sphere inputs.
See Also

OptixBuildInputSphereArray

4.12.4.4 enum OptixBuildOperation

Enum to specify the acceleration build operation.

Used in OptixAccelBuildOptions, which is then passed to optixAccelBuild and optixAccelComputeMemoryUsage, this enum indicates whether to do a build or an update of the acceleration structure.

Acceleration structure updates utilize the same acceleration structure, but with updated bounds. Updates are typically much faster than builds, however, large perturbations can degrade the quality of the acceleration structure.

See Also

optixAccelComputeMemoryUsage(), optixAccelBuild(), OptixAccelBuildOptions

Enumerator

OPTIX_BUILD_OPERATION_BUILD Perform a full build operation.

OPTIX_BUILD_OPERATION_UPDATE Perform an update using new bounds.

4.12.4.5 enum OptixCompileDebugLevel

Debug levels.

See Also

OptixModuleCompileOptions::debugLevel

Enumerator

OPTIX_COMPILE_DEBUG_LEVEL_DEFAULT Default currently is minimal.

OPTIX_COMPILE_DEBUG_LEVEL_NONE No debug information.

OPTIX_COMPILE_DEBUG_LEVEL_MINIMAL Generate information that does not impact performance. Note this replaces OPTIX_COMPILE_DEBUG_LEVEL_LINEINFO.

OPTIX_COMPILE_DEBUG_LEVEL_MODERATE Generate some debug information with slight performance cost.

OPTIX_COMPILE_DEBUG_LEVEL_FULL Generate full debug information.

4.12.4.6 enum OptixCompileOptimizationLevel

Optimization levels.

See Also

OptixModuleCompileOptions::optLevel

Enumerator

OPTIX_COMPILE_OPTIMIZATION_DEFAULT Default is to run all optimizations.

OPTIX_COMPILE_OPTIMIZATION_LEVEL_0 No optimizations.

OPTIX_COMPILE_OPTIMIZATION_LEVEL_1 Some optimizations.

OPTIX_COMPILE_OPTIMIZATION_LEVEL_2 Most optimizations.

OPTIX_COMPILE_OPTIMIZATION_LEVEL_3 All optimizations.

4.12.4.7 enum OptixCurveEndcapFlags

Curve end cap types, for non-linear curves.

Enumerator

OPTIX_CURVE_ENDCAP_DEFAULT Default end caps. Round end caps for linear, no end caps for quadratic/cubic.

OPTIX_CURVE_ENDCAP_ON Flat end caps at both ends of quadratic/cubic curve segments. Not valid for linear.

4.12.4.8 enum OptixDenoiserAlphaMode

Various parameters used by the denoiser.

See Also

optixDenoiserInvoke()
optixDenoiserComputeIntensity()
optixDenoiserComputeAverageColor()

Enumerator

OPTIX_DENOISER_ALPHA_MODE_COPY Copy alpha (if present) from input layer, no denoising.

OPTIX_DENOISER_ALPHA_MODE_ALPHA_AS_AOV Denoise alpha separately. With AOV model kinds, treat alpha like an AOV.

OPTIX_DENOISER_ALPHA_MODE_FULL_DENOISE_PASS With AOV model kinds, full denoise pass with alpha. This is slower than OPTIX_DENOISER_ALPHA_MODE_ALPHA_AS_AOV.

4.12.4.9 enum OptixDenoiserModelKind

Model kind used by the denoiser.

See Also

optixDenoiserCreate

Enumerator

OPTIX_DENOISER_MODEL_KIND_LDR Use the built-in model appropriate for low dynamic range input.

OPTIX_DENOISER_MODEL_KIND_HDR Use the built-in model appropriate for high dynamic range input.

OPTIX_DENOISER_MODEL_KIND_AOV Use the built-in model appropriate for high dynamic range input and support for AOVs.

OPTIX_DENOISER_MODEL_KIND_TEMPORAL Use the built-in model appropriate for high dynamic range input, temporally stable.

OPTIX_DENOISER_MODEL_KIND_TEMPORAL_AOV Use the built-in model appropriate for high dynamic range input and support for AOVs, temporally stable.

OPTIX_DENOISER_MODEL_KIND_UPSCALE2X Use the built-in model appropriate for high dynamic range input and support for AOVs, upscaling 2x.

OPTIX_DENOISER_MODEL_KIND_TEMPORAL_UPSCALE2X Use the built-in model appropriate for high dynamic range input and support for AOVs, upscaling 2x, temporally stable.

4.12.4.10 enum OptixDeviceContextValidationMode

Validation mode settings.

When enabled, certain device code utilities will be enabled to provide as good debug and error checking facilities as possible.

See Also

optixDeviceContextCreate()

Enumerator

OPTIX_DEVICE_CONTEXT_VALIDATION_MODE_OFF

OPTIX_DEVICE_CONTEXT_VALIDATION_MODE_ALL

4.12.4.11 enum OptixDeviceProperty

Parameters used for optixDeviceContextGetProperty()

See Also

optixDeviceContextGetProperty()

Enumerator

OPTIX_DEVICE_PROPERTY_LIMIT_MAX_TRACE_DEPTH Maximum value for OptixPipelineLinkOptions::maxTraceDepth. sizeof(unsigned int)

OPTIX_DEVICE_PROPERTY_LIMIT_MAX_TRAVERSABLE_GRAPH_DEPTH Maximum value to pass into optixPipelineSetStackSize for parameter maxTraversableGraphDepth.v sizeof(unsigned int)

OPTIX_DEVICE_PROPERTY_LIMIT_MAX_PRIMITIVES_PER_GAS The maximum number of primitives (over all build inputs) as input to a single Geometry Acceleration Structure (GAS). sizeof(unsigned int)

OPTIX_DEVICE_PROPERTY_LIMIT_MAX_INSTANCES_PER_IAS The maximum number of instances (over all build inputs) as input to a single Instance Acceleration Structure (IAS). sizeof(unsigned int)

OPTIX_DEVICE_PROPERTY_RTCORE_VERSION The RT core version supported by the device (0 for no support, 10 for version 1.0). sizeof(unsigned int)

OPTIX_DEVICE_PROPERTY_LIMIT_MAX_INSTANCE_ID The maximum value for OptixInstance::instanceId. sizeof(unsigned int)

OPTIX_DEVICE_PROPERTY_LIMIT_NUM_BITS_INSTANCE_VISIBILITY_MASK The number of bits available for the OptixInstance::visibilityMask. Higher bits must be set to zero. sizeof(unsigned int)

OPTIX_DEVICE_PROPERTY_LIMIT_MAX_SBT_RECORDS_PER_GAS The maximum number of instances that can be added to a single Instance Acceleration Structure (IAS). sizeof(unsigned int)

OPTIX_DEVICE_PROPERTY_LIMIT_MAX_SBT_OFFSET The maximum value for OptixInstance::sbtOffset. sizeof(unsigned int)

4.12.4.12 enum OptixExceptionCodes

The following values are used to indicate which exception was thrown.

Enumerator

OPTIX_EXCEPTION_CODE_STACK_OVERFLOW Stack overflow of the continuation stack. no exception details.

OPTIX_EXCEPTION_CODE_TRACE_DEPTH_EXCEEDED The trace depth is exceeded. no exception details.

OPTIX_EXCEPTION_CODE_TRAVERSAL_DEPTH_EXCEEDED The traversal depth is exceeded. Exception details: optixGetTransformListSize() optixGetTransformListHandle()

OPTIX_EXCEPTION_CODE_TRAVERSAL_INVALID_TRAVERSABLE Traversal encountered an invalid traversable type. Exception details: optixGetTransformListSize() optixGetTransformListHandle() optixGetExceptionInvalidTraversable()

OPTIX_EXCEPTION_CODE_TRAVERSAL_INVALID_MISS_SBT The miss SBT record index is out of bounds A miss SBT record index is valid within the range [0, OptixShaderBindingTable::missRecordCount) (See optixLaunch) Exception details: optixGetExceptionInvalidSbtOffset()

OPTIX_EXCEPTION_CODE_TRAVERSAL_INVALID_HIT_SBT The traversal hit SBT record index out of bounds. A traversal hit SBT record index is valid within the range [0, OptixShaderBindingTable::hitgroupRecordCount) (See optixLaunch) The following formula relates the sbt-geometry-acceleration-structure-index (See optixGetSbtGASIndex), sbt-stride-from-trace-call and sbt-offset-from-trace-call (See optixTrace)

sbt-index = sbt-instance-offset + (sbt-geometry-acceleration-structure-index $*$ sbt-stride-from-trace-call) + sbt-offset-from-trace-call

Exception details: optixGetTransformListSize() optixGetTransformListHandle() optixGetExceptionInvalidSbtOffset() optixGetSbtGASIndex()

OPTIX_EXCEPTION_CODE_UNSUPPORTED_PRIMITIVE_TYPE The shader encountered an unsupported primitive type (See OptixPipelineCompileOptions::usesPrimitiveTypeFlags). no exception details.

OPTIX_EXCEPTION_CODE_INVALID_RAY The shader encountered a call to optixTrace with at least one of the float arguments being inf or nan, or the tmin argument is negative. Exception details: optixGetExceptionInvalidRay()

OPTIX_EXCEPTION_CODE_CALLABLE_PARAMETER_MISMATCH The shader encountered a call to either optixDirectCall or optixCallableCall where the argument count does not match the parameter count of the callable program which is called. Exception details: optixGetExceptionParameterMismatch.

OPTIX_EXCEPTION_CODE_BUILTIN_IS_MISMATCH The invoked builtin IS does not match the current GAS.

OPTIX_EXCEPTION_CODE_CALLABLE_INVALID_SBT Tried to call a callable program using an SBT offset that is larger than the number of passed in callable SBT records. Exception details: optixGetExceptionInvalidSbtOffset()

OPTIX_EXCEPTION_CODE_CALLABLE_NO_DC_SBT_RECORD Tried to call a direct callable using an SBT offset of a record that was built from a program group that did not include a direct callable.

OPTIX_EXCEPTION_CODE_CALLABLE_NO_CC_SBT_RECORD Tried to call a continuation callable using an SBT offset of a record that was built from a program group that did not include a continuation callable.

OPTIX_EXCEPTION_CODE_UNSUPPORTED_SINGLE_LEVEL_GAS Tried to directly traverse a single gas while single gas traversable graphs are not enabled (see OptixTraversable-GraphFlags::OPTIX_TRAVERSABLE_GRAPH_FLAG_ALLOW_SINGLE_GAS). Exception details: optixGetTransformListSize() optixGetTransformListHandle() optixGetExceptionInvalidTraversable()

OPTIX_EXCEPTION_CODE_INVALID_VALUE_ARGUMENT_0 argument passed to an optix call is not within an acceptable range of values.

OPTIX_EXCEPTION_CODE_INVALID_VALUE_ARGUMENT_1

OPTIX_EXCEPTION_CODE_INVALID_VALUE_ARGUMENT_2

OPTIX_EXCEPTION_CODE_UNSUPPORTED_DATA_ACCESS Tried to access data on an AS without random data access support (See OptixBuildFlags).

OPTIX_EXCEPTION_CODE_PAYLOAD_TYPE_MISMATCH The program payload type doesn't match the trace payload type.

4.12.4.13 enum OptixExceptionFlags

Exception flags.

See Also

OptixPipelineCompileOptions::exceptionFlags, OptixExceptionCodes

Enumerator

OPTIX_EXCEPTION_FLAG_NONE No exception are enabled.

OPTIX_EXCEPTION_FLAG_STACK_OVERFLOW Enables exceptions check related to the continuation stack.

OPTIX_EXCEPTION_FLAG_TRACE_DEPTH Enables exceptions check related to trace depth.

OPTIX_EXCEPTION_FLAG_USER Enables user exceptions via optixThrowException(). This flag must be specified for all modules in a pipeline if any module calls optixThrowException().

OPTIX_EXCEPTION_FLAG_DEBUG Enables various exceptions check related to traversal.

4.12.4.14 enum OptixGeometryFlags

Flags used by OptixBuildInputTriangleArray::flags and #OptixBuildInput::flag and OptixBuildInputCustomPrimitiveArray::flags.

Enumerator

OPTIX_GEOMETRY_FLAG_NONE No flags set.

OPTIX_GEOMETRY_FLAG_DISABLE_ANYHIT Disables the invocation of the anyhit program. Can be overridden by OPTIX_INSTANCE_FLAG_ENFORCE_ANYHIT and OPTIX_RAY_FLAG_ENFORCE_ANYHIT.

OPTIX_GEOMETRY_FLAG_REQUIRE_SINGLE_ANYHIT_CALL If set, an intersection with the primitive will trigger one and only one invocation of the anyhit program. Otherwise, the anyhit program may be invoked more than once.

OPTIX_GEOMETRY_FLAG_DISABLE_TRIANGLE_FACE_CULLING Prevent triangles from getting culled due to their orientation. Effectively ignores ray flags OPTIX_RAY_FLAG_CULL_BACK_FACING_TRIANGLES and OPTIX_RAY_FLAG_CULL_FRONT_FACING_TRIANGLES.

4.12.4.15 enum OptixHitKind

Legacy type: A subset of the hit kinds for built-in primitive intersections. It is preferred to use optixGetPrimitiveType(), together with optixIsFrontFaceHit() or optixIsBackFaceHit().

See Also

optixGetHitKind()

Enumerator

OPTIX_HIT_KIND_TRIANGLE_FRONT_FACE Ray hit the triangle on the front face.

OPTIX_HIT_KIND_TRIANGLE_BACK_FACE Ray hit the triangle on the back face.

4.12.4.16 enum OptixIndicesFormat

Format of indices used int OptixBuildInputTriangleArray::indexFormat.

Enumerator

OPTIX_INDICES_FORMAT_NONE No indices, this format must only be used in combination with triangle soups, i.e., numIndexTriplets must be zero.

OPTIX_INDICES_FORMAT_UNSIGNED_SHORT3 Three shorts.

OPTIX_INDICES_FORMAT_UNSIGNED_INT3 Three ints.

4.12.4.17 enum OptixInstanceFlags

Flags set on the OptixInstance::flags.

These can be or'ed together to combine multiple flags.

Enumerator

OPTIX_INSTANCE_FLAG_NONE No special flag set.

OPTIX_INSTANCE_FLAG_DISABLE_TRIANGLE_FACE_CULLING Prevent triangles from getting culled due to their orientation. Effectively ignores ray flags OPTIX_RAY_FLAG_CULL_BACK_FACING_TRIANGLES and OPTIX_RAY_FLAG_CULL_FRONT_FACING_TRIANGLES.

OPTIX_INSTANCE_FLAG_FLIP_TRIANGLE_FACING Flip triangle orientation. This affects front/backface culling as well as the reported face in case of a hit.

OPTIX_INSTANCE_FLAG_DISABLE_ANYHIT Disable anyhit programs for all geometries of the instance. Can be overridden by OPTIX_RAY_FLAG_ENFORCE_ANYHIT. This flag is mutually exclusive with OPTIX_INSTANCE_FLAG_ENFORCE_ANYHIT.

OPTIX_INSTANCE_FLAG_ENFORCE_ANYHIT Enables anyhit programs for all geometries of the instance. Overrides OPTIX_GEOMETRY_FLAG_DISABLE_ANYHIT Can be overridden by OPTIX_RAY_FLAG_DISABLE_ANYHIT. This flag is mutually exclusive with OPTIX_INSTANCE_FLAG_DISABLE_ANYHIT.

4.12.4.18 enum OptixModuleCompileState

Module compilation state.

See Also

optixModuleGetCompilationState(), optixModuleCreateFromPTXWithTasks()

Enumerator

OPTIX_MODULE_COMPILE_STATE_NOT_STARTED No OptixTask objects have started.

OPTIX_MODULE_COMPILE_STATE_STARTED Started, but not all OptixTask objects have completed. No detected failures.

OPTIX_MODULE_COMPILE_STATE_IMPENDING_FAILURE Not all OptixTask objects have completed, but at least one has failed.

OPTIX_MODULE_COMPILE_STATE_FAILED All OptixTask objects have completed, and at least one has failed.

OPTIX_MODULE_COMPILE_STATE_COMPLETED All OptixTask objects have completed. The OptixModule is ready to be used.

4.12.4.19 enum OptixMotionFlags

Enum to specify motion flags.

See Also

OptixMotionOptions::flags.

Enumerator

OPTIX_MOTION_FLAG_NONE

OPTIX_MOTION_FLAG_START_VANISH

OPTIX_MOTION_FLAG_END_VANISH

4.12.4.20 enum OptixPayloadSemantics

Semantic flags for a single payload word.

Used to specify the semantics of a payload word per shader type. "read": Shader of this type may read the payload word. "write": Shader of this type may write the payload word.

"trace_caller_write": Shaders may consume the value of the payload word passed to optixTrace by the caller. "trace_caller_read": The caller to optixTrace may read the payload word after the call to optixTrace.

Semantics can be bitwise combined. Combining "read" and "write" is equivalent to specifying "read_write". A payload needs to be writable by the caller or at least one shader type. A payload needs to be readable by the caller or at least one shader type after a being writable.

Enumerator

OPTIX_PAYLOAD_SEMANTICS_TRACE_CALLER_NONE

OPTIX_PAYLOAD_SEMANTICS_TRACE_CALLER_READ

OPTIX_PAYLOAD_SEMANTICS_TRACE_CALLER_WRITE

OPTIX_PAYLOAD_SEMANTICS_TRACE_CALLER_READ_WRITE

OPTIX_PAYLOAD_SEMANTICS_CH_NONE

OPTIX_PAYLOAD_SEMANTICS_CH_READ

OPTIX_PAYLOAD_SEMANTICS_CH_WRITE

OPTIX_PAYLOAD_SEMANTICS_CH_READ_WRITE

OPTIX_PAYLOAD_SEMANTICS_MS_NONE

OPTIX_PAYLOAD_SEMANTICS_MS_READ

OPTIX_PAYLOAD_SEMANTICS_MS_WRITE

OPTIX_PAYLOAD_SEMANTICS_MS_READ_WRITE

OPTIX_PAYLOAD_SEMANTICS_AH_NONE

OPTIX_PAYLOAD_SEMANTICS_AH_READ

OPTIX_PAYLOAD_SEMANTICS_AH_WRITE

OPTIX_PAYLOAD_SEMANTICS_AH_READ_WRITE

OPTIX_PAYLOAD_SEMANTICS_IS_NONE

OPTIX_PAYLOAD_SEMANTICS_IS_READ

OPTIX_PAYLOAD_SEMANTICS_IS_WRITE

OPTIX_PAYLOAD_SEMANTICS_IS_READ_WRITE

4.12.4.21 enum OptixPayloadTypeID

Payload type identifiers.

Enumerator

OPTIX_PAYLOAD_TYPE_DEFAULT

OPTIX_PAYLOAD_TYPE_ID_0

OPTIX_PAYLOAD_TYPE_ID_1

OPTIX_PAYLOAD_TYPE_ID_2

OPTIX_PAYLOAD_TYPE_ID_3

OPTIX_PAYLOAD_TYPE_ID_4

OPTIX_PAYLOAD_TYPE_ID_5

OPTIX_PAYLOAD_TYPE_ID_6

OPTIX_PAYLOAD_TYPE_ID_7

4.12.4.22 enum OptixPixelFormat

Pixel formats used by the denoiser.

See Also

OptixImage2D::format

Enumerator

OPTIX_PIXEL_FORMAT_HALF2 two halfs, XY

OPTIX_PIXEL_FORMAT_HALF3 three halfs, RGB

OPTIX_PIXEL_FORMAT_HALF4 four halfs, RGBA

OPTIX_PIXEL_FORMAT_FLOAT2 two floats, XY

OPTIX_PIXEL_FORMAT_FLOAT3 three floats, RGB

OPTIX_PIXEL_FORMAT_FLOAT4 four floats, RGBA

OPTIX_PIXEL_FORMAT_UCHAR3 three unsigned chars, RGB

OPTIX_PIXEL_FORMAT_UCHAR4 four unsigned chars, RGBA

OPTIX_PIXEL_FORMAT_INTERNAL_GUIDE_LAYER internal format

4.12.4.23 enum OptixPrimitiveType

Builtin primitive types.

Enumerator

OPTIX_PRIMITIVE_TYPE_CUSTOM Custom primitive.

OPTIX_PRIMITIVE_TYPE_ROUND_QUADRATIC_BSPLINE B-spline curve of degree 2 with circular cross-section.

OPTIX_PRIMITIVE_TYPE_ROUND_CUBIC_BSPLINE B-spline curve of degree 3 with circular cross-section.

OPTIX_PRIMITIVE_TYPE_ROUND_LINEAR Piecewise linear curve with circular cross-section.

OPTIX_PRIMITIVE_TYPE_ROUND_CATMULLROM CatmullRom curve with circular cross-section.

OPTIX_PRIMITIVE_TYPE_SPHERE

OPTIX_PRIMITIVE_TYPE_TRIANGLE Triangle.

4.12.4.24 enum OptixPrimitiveTypeFlags

Builtin flags may be bitwise combined.

See Also

OptixPipelineCompileOptions::usesPrimitiveTypeFlags

Enumerator

OPTIX_PRIMITIVE_TYPE_FLAGS_CUSTOM Custom primitive.

OPTIX_PRIMITIVE_TYPE_FLAGS_ROUND_QUADRATIC_BSPLINE B-spline curve of degree 2 with circular cross-section.

OPTIX_PRIMITIVE_TYPE_FLAGS_ROUND_CUBIC_BSPLINE B-spline curve of degree 3 with circular cross-section.

OPTIX_PRIMITIVE_TYPE_FLAGS_ROUND_LINEAR Piecewise linear curve with circular cross-section.

OPTIX_PRIMITIVE_TYPE_FLAGS_ROUND_CATMULLROM CatmullRom curve with circular cross-section.

OPTIX_PRIMITIVE_TYPE_FLAGS_SPHERE

OPTIX_PRIMITIVE_TYPE_FLAGS_TRIANGLE Triangle.

4.12.4.25 enum OptixProgramGroupFlags

Flags for program groups.

Enumerator

OPTIX_PROGRAM_GROUP_FLAGS_NONE Currently there are no flags.

4.12.4.26 enum OptixProgramGroupKind

Distinguishes different kinds of program groups.

Enumerator

OPTIX_PROGRAM_GROUP_KIND_RAYGEN Program group containing a raygen (RG) program.
See Also

OptixProgramGroupSingleModule, OptixProgramGroupDesc::raygen

OPTIX_PROGRAM_GROUP_KIND_MISS Program group containing a miss (MS) program.
See Also

OptixProgramGroupSingleModule, OptixProgramGroupDesc::miss

OPTIX_PROGRAM_GROUP_KIND_EXCEPTION Program group containing an exception (EX) program.
See Also

OptixProgramGroupHitgroup, OptixProgramGroupDesc::exception

OPTIX_PROGRAM_GROUP_KIND_HITGROUP Program group containing an intersection (IS), any hit (AH), and/or closest hit (CH) program.
See Also

OptixProgramGroupSingleModule, OptixProgramGroupDesc::hitgroup

OPTIX_PROGRAM_GROUP_KIND_CALLABLES Program group containing a direct (DC) or continuation (CC) callable program.
See Also

OptixProgramGroupCallables, OptixProgramGroupDesc::callables

4.12.4.27 enum OptixQueryFunctionTableOptions

Options that can be passed to `optixQueryFunctionTable()`

Enumerator

OPTIX_QUERY_FUNCTION_TABLE_OPTION_DUMMY Placeholder (there are no options yet)

4.12.4.28 enum OptixRayFlags

Ray flags passed to the device function optixTrace(). These affect the behavior of traversal per invocation.

See Also

optixTrace()

Enumerator

OPTIX_RAY_FLAG_NONE No change from the behavior configured for the individual AS.

OPTIX_RAY_FLAG_DISABLE_ANYHIT Disables anyhit programs for the ray. Overrides OPTIX_INSTANCE_FLAG_ENFORCE_ANYHIT. This flag is mutually exclusive with OPTIX_RAY_FLAG_ENFORCE_ANYHIT, OPTIX_RAY_FLAG_CULL_DISABLED_ANYHIT, OPTIX_RAY_FLAG_CULL_ENFORCED_ANYHIT.

OPTIX_RAY_FLAG_ENFORCE_ANYHIT Forces anyhit program execution for the ray. Overrides OPTIX_GEOMETRY_FLAG_DISABLE_ANYHIT as well as OPTIX_INSTANCE_FLAG_DISABLE_ANYHIT. This flag is mutually exclusive with OPTIX_RAY_FLAG_DISABLE_ANYHIT, OPTIX_RAY_FLAG_CULL_DISABLED_ANYHIT, OPTIX_RAY_FLAG_CULL_ENFORCED_ANYHIT.

OPTIX_RAY_FLAG_TERMINATE_ON_FIRST_HIT Terminates the ray after the first hit and executes the closesthit program of that hit.

OPTIX_RAY_FLAG_DISABLE_CLOSESTHIT Disables closesthit programs for the ray, but still executes miss program in case of a miss.

OPTIX_RAY_FLAG_CULL_BACK_FACING_TRIANGLES Do not intersect triangle back faces (respects a possible face change due to instance flag OPTIX_INSTANCE_FLAG_FLIP_TRIANGLE_FACING). This flag is mutually exclusive with OPTIX_RAY_FLAG_CULL_FRONT_FACING_TRIANGLES.

OPTIX_RAY_FLAG_CULL_FRONT_FACING_TRIANGLES Do not intersect triangle front faces (respects a possible face change due to instance flag OPTIX_INSTANCE_FLAG_FLIP_TRIANGLE_FACING). This flag is mutually exclusive with OPTIX_RAY_FLAG_CULL_BACK_FACING_TRIANGLES.

OPTIX_RAY_FLAG_CULL_DISABLED_ANYHIT Do not intersect geometry which disables anyhit programs (due to setting geometry flag OPTIX_GEOMETRY_FLAG_DISABLE_ANYHIT or instance flag OPTIX_INSTANCE_FLAG_DISABLE_ANYHIT). This flag is mutually exclusive with OPTIX_RAY_FLAG_CULL_ENFORCED_ANYHIT, OPTIX_RAY_FLAG_ENFORCE_ANYHIT, OPTIX_RAY_FLAG_DISABLE_ANYHIT.

OPTIX_RAY_FLAG_CULL_ENFORCED_ANYHIT Do not intersect geometry which have an enabled anyhit program (due to not setting geometry flag

OPTIX_GEOMETRY_FLAG_DISABLE_ANYHIT or setting instance flag OPTIX_INSTANCE_FLAG_ENFORCE_ANYHIT). This flag is mutually exclusive with OPTIX_RAY_FLAG_CULL_DISABLED_ANYHIT, OPTIX_RAY_FLAG_ENFORCE_ANYHIT, OPTIX_RAY_FLAG_DISABLE_ANYHIT.

4.12.4.29 enum OptixResult

Result codes returned from API functions.

All host side API functions return OptixResult with the exception of optixGetErrorName and optixGetErrorString. When successful OPTIX_SUCCESS is returned. All return codes except for OPTIX_SUCCESS should be assumed to be errors as opposed to a warning.

See Also

optixGetErrorName(), optixGetErrorString()

Enumerator

OPTIX_SUCCESS

OPTIX_ERROR_INVALID_VALUE

OPTIX_ERROR_HOST_OUT_OF_MEMORY

OPTIX_ERROR_INVALID_OPERATION

OPTIX_ERROR_FILE_IO_ERROR

OPTIX_ERROR_INVALID_FILE_FORMAT

OPTIX_ERROR_DISK_CACHE_INVALID_PATH

OPTIX_ERROR_DISK_CACHE_PERMISSION_ERROR

OPTIX_ERROR_DISK_CACHE_DATABASE_ERROR

OPTIX_ERROR_DISK_CACHE_INVALID_DATA

OPTIX_ERROR_LAUNCH_FAILURE

OPTIX_ERROR_INVALID_DEVICE_CONTEXT

OPTIX_ERROR_CUDA_NOT_INITIALIZED

OPTIX_ERROR_VALIDATION_FAILURE

OPTIX_ERROR_INVALID_PTX

OPTIX_ERROR_INVALID_LAUNCH_PARAMETER

OPTIX_ERROR_INVALID_PAYLOAD_ACCESS

OPTIX_ERROR_INVALID_ATTRIBUTE_ACCESS

OPTIX_ERROR_INVALID_FUNCTION_USE

OPTIX_ERROR_INVALID_FUNCTION_ARGUMENTS

OPTIX_ERROR_PIPELINE_OUT_OF_CONSTANT_MEMORY

OPTIX_ERROR_PIPELINE_LINK_ERROR

OPTIX_ERROR_ILLEGAL_DURING_TASK_EXECUTE

OPTIX_ERROR_INTERNAL_COMPILER_ERROR

OPTIX_ERROR_DENOISER_MODEL_NOT_SET

OPTIX_ERROR_DENOISER_NOT_INITIALIZED

OPTIX_ERROR_ACCEL_NOT_COMPATIBLE

OPTIX_ERROR_PAYLOAD_TYPE_MISMATCH

OPTIX_ERROR_PAYLOAD_TYPE_RESOLUTION_FAILED

OPTIX_ERROR_PAYLOAD_TYPE_ID_INVALID

OPTIX_ERROR_NOT_SUPPORTED

OPTIX_ERROR_UNSUPPORTED_ABI_VERSION

OPTIX_ERROR_FUNCTION_TABLE_SIZE_MISMATCH

OPTIX_ERROR_INVALID_ENTRY_FUNCTION_OPTIONS

OPTIX_ERROR_LIBRARY_NOT_FOUND

OPTIX_ERROR_ENTRY_SYMBOL_NOT_FOUND

OPTIX_ERROR_LIBRARY_UNLOAD_FAILURE

OPTIX_ERROR_DEVICE_OUT_OF_MEMORY

OPTIX_ERROR_CUDA_ERROR

OPTIX_ERROR_INTERNAL_ERROR

OPTIX_ERROR_UNKNOWN

4.12.4.30 enum OptixTransformFormat

Format of transform used in OptixBuildInputTriangleArray::transformFormat.

Enumerator

OPTIX_TRANSFORM_FORMAT_NONE no transform, default for zero initialization

OPTIX_TRANSFORM_FORMAT_MATRIX_FLOAT12 3x4 row major affine matrix

4.12.4.31 enum OptixTransformType

Transform.

OptixTransformType is used by the device function optixGetTransformTypeFromHandle() to determine the type of the OptixTraversableHandle returned from optixGetTransformListHandle().

Enumerator

OPTIX_TRANSFORM_TYPE_NONE Not a transformation.
See Also

OPTIX_TRANSFORM_TYPE_STATIC_TRANSFORM OptixStaticTransform
See Also

OPTIX_TRANSFORM_TYPE_MATRIX_MOTION_TRANSFORM
OptixMatrixMotionTransform
See Also

OPTIX_TRANSFORM_TYPE_SRT_MOTION_TRANSFORM OptixSRTMotionTransform
See Also

OPTIX_TRANSFORM_TYPE_INSTANCE OptixInstance

4.12.4.32 enum OptixTraversableGraphFlags

Specifies the set of valid traversable graphs that may be passed to invocation of optixTrace(). Flags may be bitwise combined.

Enumerator

OPTIX_TRAVERSABLE_GRAPH_FLAG_ALLOW_ANY Used to signal that any traversable graphs is valid. This flag is mutually exclusive with all other flags.

OPTIX_TRAVERSABLE_GRAPH_FLAG_ALLOW_SINGLE_GAS Used to signal that a traversable graph of a single Geometry Acceleration Structure (GAS) without any transforms is valid. This flag may be combined with other flags except for OPTIX_TRAVERSABLE_GRAPH_FLAG_ALLOW_ANY.

OPTIX_TRAVERSABLE_GRAPH_FLAG_ALLOW_SINGLE_LEVEL_INSTANCING Used to signal that a traversable graph of a single Instance Acceleration Structure (IAS) directly connected to Geometry Acceleration Structure (GAS) traversables without transform traversables in between is valid. This flag may be combined with other flags except for OPTIX_TRAVERSABLE_GRAPH_FLAG_ALLOW_ANY.

4.12.4.33 enum OptixTraversableType

Traversable Handles.

See Also

optixConvertPointerToTraversableHandle()

Enumerator

OPTIX_TRAVERSABLE_TYPE_STATIC_TRANSFORM Static transforms.
See Also

OptixStaticTransform

OPTIX_TRAVERSABLE_TYPE_MATRIX_MOTION_TRANSFORM Matrix motion transform.
See Also

OptixMatrixMotionTransform

OPTIX_TRAVERSABLE_TYPE_SRT_MOTION_TRANSFORM SRT motion transform.
See Also

OptixSRTMotionTransform

4.12.4.34 enum OptixVertexFormat

Format of vertices used in OptixBuildInputTriangleArray::vertexFormat.

Enumerator

OPTIX_VERTEX_FORMAT_NONE No vertices.

OPTIX_VERTEX_FORMAT_FLOAT3 Vertices are represented by three floats.

OPTIX_VERTEX_FORMAT_FLOAT2 Vertices are represented by two floats.

OPTIX_VERTEX_FORMAT_HALF3 Vertices are represented by three halfs.

OPTIX_VERTEX_FORMAT_HALF2 Vertices are represented by two halfs.

OPTIX_VERTEX_FORMAT_SNORM16_3

OPTIX_VERTEX_FORMAT_SNORM16_2

4.13 Function Table

Classes

- struct OptixFunctionTable

Typedefs

- typedef struct OptixFunctionTable OptixFunctionTable

Variables

- OptixFunctionTable g_optixFunctionTable

4.13.1 Detailed Description

OptiX Function Table.

4.13.2 Typedef Documentation

4.13.2.1 typedef struct OptixFunctionTable OptixFunctionTable

The function table containing all API functions.

See optixInit() and optixInitWithHandle().

4.13.3 Variable Documentation

4.13.3.1 OptixFunctionTable g_optixFunctionTable

If the stubs in optix_stubs.h are used, then the function table needs to be defined in exactly one translation unit. This can be achieved by including this header file in that translation unit.

4.14 Utilities

Classes

- struct OptixUtilDenoiserImageTile

Functions

- OptixResult optixUtilAccumulateStackSizes (OptixProgramGroup programGroup, OptixStackSizes ∗stackSizes)
- OptixResult optixUtilComputeStackSizes (const OptixStackSizes ∗stackSizes, unsigned int maxTraceDepth, unsigned int maxCCDepth, unsigned int maxDCDepth, unsigned int ∗directCallableStackSizeFromTraversal, unsigned int ∗directCallableStackSizeFromState, unsigned int ∗continuationStackSize)
- OptixResult optixUtilComputeStackSizesDCSplit (const OptixStackSizes ∗stackSizes, unsigned int dssDCFromTraversal, unsigned int dssDCFromState, unsigned int maxTraceDepth, unsigned int maxCCDepth, unsigned int maxDCDepthFromTraversal, unsigned int maxDCDepthFromState, unsigned int ∗directCallableStackSizeFromTraversal, unsigned int ∗directCallableStackSizeFromState, unsigned int ∗continuationStackSize)
- OptixResult optixUtilComputeStackSizesCssCCTree (const OptixStackSizes ∗stackSizes, unsigned int cssCCTree, unsigned int maxTraceDepth, unsigned int maxDCDepth, unsigned int ∗directCallableStackSizeFromTraversal, unsigned int ∗directCallableStackSizeFromState, unsigned int ∗continuationStackSize)
- OptixResult optixUtilComputeStackSizesSimplePathTracer (OptixProgramGroup programGroupRG, OptixProgramGroup programGroupMS1, const OptixProgramGroup ∗programGroupCH1, unsigned int programGroupCH1Count, OptixProgramGroup programGroupMS2, const OptixProgramGroup ∗programGroupCH2, unsigned int programGroupCH2Count, unsigned int ∗directCallableStackSizeFromTraversal, unsigned int ∗directCallableStackSizeFromState, unsigned int ∗continuationStackSize)
- OptixResult optixUtilGetPixelStride (const OptixImage2D &image, unsigned int &pixelStrideInBytes)
- OptixResult optixUtilDenoiserSplitImage (const OptixImage2D &input, const OptixImage2D &output, unsigned int overlapWindowSizeInPixels, unsigned int tileWidth, unsigned int tileHeight, std::vector< OptixUtilDenoiserImageTile > &tiles)
- OptixResult optixUtilDenoiserInvokeTiled (OptixDenoiser denoiser, CUstream stream, const OptixDenoiserParams ∗params, CUdeviceptr denoiserState, size_t denoiserStateSizeInBytes, const OptixDenoiserGuideLayer ∗guideLayer, const OptixDenoiserLayer ∗layers, unsigned int numLayers, CUdeviceptr scratch, size_t scratchSizeInBytes, unsigned int overlapWindowSizeInPixels, unsigned int tileWidth, unsigned int tileHeight)
- OptixResult optixInitWithHandle (void ∗∗handlePtr)
- OptixResult optixInit (void)
- OptixResult optixUninitWithHandle (void ∗handle)

4.14.1 Detailed Description

OptiX Utilities.

4.14.2 Function Documentation

4.14.2.1 OptixResult optixInit (void) [inline]

Loads the OptiX library and initializes the function table used by the stubs below.

A variant of optixInitWithHandle() that does not make the handle to the loaded library available.

4.14.2.2 OptixResult optixInitWithHandle (void ** *handlePtr*) [inline]

Loads the OptiX library and initializes the function table used by the stubs below.

If handlePtr is not nullptr, an OS-specific handle to the library will be returned in *handlePtr.

See Also

optixUninitWithHandle

4.14.2.3 OptixResult optixUninitWithHandle (void * *handle*) [inline]

Unloads the OptiX library and zeros the function table used by the stubs below. Takes the handle returned by optixInitWithHandle. All OptixDeviceContext objects must be destroyed before calling this function, or the behavior is undefined.

See Also

optixInitWithHandle

4.14.2.4 OptixResult optixUtilAccumulateStackSizes (OptixProgramGroup *programGroup*, OptixStackSizes * *stackSizes*) [inline]

Retrieves direct and continuation stack sizes for each program in the program group and accumulates the upper bounds in the correponding output variables based on the semantic type of the program. Before the first invocation of this function with a given instance of OptixStackSizes, the members of that instance should be set to 0.

4.14.2.5 OptixResult optixUtilComputeStackSizes (const OptixStackSizes * *stackSizes*, unsigned int *maxTraceDepth*, unsigned int *maxCCDepth*, unsigned int *maxDCDepth*, unsigned int * *directCallableStackSizeFromTraversal*, unsigned int * *directCallableStackSizeFromState*, unsigned int * *continuationStackSize*) [inline]

Computes the stack size values needed to configure a pipeline.

See the programming guide for an explanation of the formula.

Parameters

in	*stackSizes*	Accumulated stack sizes of all programs in the call graph.
in	*maxTraceDepth*	Maximum depth of optixTrace() calls.
in	*maxCCDepth*	Maximum depth of calls trees of continuation callables.
in	*maxDCDepth*	Maximum depth of calls trees of direct callables.
out	*directCallableStackSizeFromTraversal*	Direct stack size requirement for direct callables invoked from IS or AH.
out	*directCallableStackSizeFromState*	Direct stack size requirement for direct callables invoked from RG, MS, or CH.
out	*continuationStackSize*	Continuation stack requirement.

4.14.2.6 OptixResult optixUtilComputeStackSizesCssCCTree (const OptixStackSizes ∗ *stackSizes*, unsigned int *cssCCTree*, unsigned int *maxTraceDepth*, unsigned int *maxDCDepth*, unsigned int ∗ *directCallableStackSizeFromTraversal*, unsigned int ∗ *directCallableStackSizeFromState*, unsigned int ∗ *continuationStackSize*) `[inline]`

Computes the stack size values needed to configure a pipeline.

This variant is similar to optixUtilComputeStackSizes(), except that it expects the value cssCCTree instead of cssCC and maxCCDepth.

See programming guide for an explanation of the formula.

Parameters

in	*stackSizes*	Accumulated stack sizes of all programs in the call graph.
in	*cssCCTree*	Maximum stack size used by calls trees of continuation callables.
in	*maxTraceDepth*	Maximum depth of optixTrace() calls.
in	*maxDCDepth*	Maximum depth of calls trees of direct callables.
out	*directCallableStackSizeFromTraversal*	Direct stack size requirement for direct callables invoked from IS or AH.
out	*directCallableStackSizeFromState*	Direct stack size requirement for direct callables invoked from RG, MS, or CH.
out	*continuationStackSize*	Continuation stack requirement.

4.14.2.7 OptixResult optixUtilComputeStackSizesDCSplit (const OptixStackSizes ∗ *stackSizes*, unsigned int *dssDCFromTraversal*, unsigned int *dssDCFromState*, unsigned int *maxTraceDepth*, unsigned int *maxCCDepth*, unsigned int *maxDCDepthFromTraversal*, unsigned int *maxDCDepthFromState*, unsigned int ∗ *directCallableStackSizeFromTraversal*, unsigned int ∗ *directCallableStackSizeFromState*, unsigned int ∗ *continuationStackSize*) `[inline]`

Computes the stack size values needed to configure a pipeline.

This variant is similar to optixUtilComputeStackSizes(), except that it expects the values dssDC and maxDCDepth split by call site semantic.

See programming guide for an explanation of the formula.

Parameters

`in`	*stackSizes*	Accumulated stack sizes of all programs in the call graph.
`in`	*dssDCFromTraversal*	Accumulated direct stack size of all DC programs invoked from IS or AH.
`in`	*dssDCFromState*	Accumulated direct stack size of all DC programs invoked from RG, MS, or CH.
`in`	*maxTraceDepth*	Maximum depth of optixTrace() calls.
`in`	*maxCCDepth*	Maximum depth of calls trees of continuation callables.
`in`	*maxDCDepthFromTraversal*	Maximum depth of calls trees of direct callables invoked from IS or AH.
`in`	*maxDCDepthFromState*	Maximum depth of calls trees of direct callables invoked from RG, MS, or CH.
`out`	*directCallableStackSizeFromTraversal*	Direct stack size requirement for direct callables invoked from IS or AH.
`out`	*directCallableStackSizeFromState*	Direct stack size requirement for direct callables invoked from RG, MS, or CH.
`out`	*continuationStackSize*	Continuation stack requirement.

4.14.2.8 OptixResult optixUtilComputeStackSizesSimplePathTracer (OptixProgramGroup *programGroupRG*, OptixProgramGroup *programGroupMS1*, const OptixProgramGroup ∗ *programGroupCH1*, unsigned int *programGroupCH1Count*,

OptixProgramGroup *programGroupMS2,*
const OptixProgramGroup * *programGroupCH2,*
unsigned int *programGroupCH2Count,*
unsigned int * *directCallableStackSizeFromTraversal,*
unsigned int * *directCallableStackSizeFromState,*
unsigned int * *continuationStackSize*) `[inline]`

Computes the stack size values needed to configure a pipeline.

This variant is a specialization of optixUtilComputeStackSizes() for a simple path tracer with the following assumptions: There are only two ray types, camera rays and shadow rays. There are only RG, MS, and CH programs, and no AH, IS, CC, or DC programs. The camera rays invoke only the miss and closest hit programs MS1 and CH1, respectively. The CH1 program might trace shadow rays, which invoke only the miss and closest hit programs MS2 and CH2, respectively.

For flexibility, we allow for each of CH1 and CH2 not just one single program group, but an array of programs groups, and compute the maximas of the stack size requirements per array.

See programming guide for an explanation of the formula.

4.14.2.9 OptixResult optixUtilDenoiserInvokeTiled (

OptixDenoiser *denoiser,*
CUstream *stream,*
const OptixDenoiserParams * *params,*
CUdeviceptr *denoiserState,*
size_t *denoiserStateSizeInBytes,*
const OptixDenoiserGuideLayer * *guideLayer,*
const OptixDenoiserLayer * *layers,*
unsigned int *numLayers,*
CUdeviceptr *scratch,*
size_t *scratchSizeInBytes,*
unsigned int *overlapWindowSizeInPixels,*
unsigned int *tileWidth,*
unsigned int *tileHeight*) `[inline]`

Run denoiser on input layers see optixDenoiserInvoke additional parameters:

Runs the denoiser on the input layers on a single GPU and stream using optixDenoiserInvoke. If the input layers' dimensions are larger than the specified tile size, the image is divided into tiles using optixUtilDenoiserSplitImage, and multiple back-to-back invocations are performed in order to reuse the scratch space. Multiple tiles can be invoked concurrently if optixUtilDenoiserSplitImage is used directly and multiple scratch allocations for each concurrent invocation are used. The input parameters are the same as optixDenoiserInvoke except for the addition of the maximum tile size.

Parameters

`in`	*denoiser*	
`in`	*stream*	

Parameters

in	*params*	
in	*denoiserState*	
in	*denoiserStateSizeInBytes*	
in	*guideLayer*	
in	*layers*	
in	*numLayers*	
in	*scratch*	
in	*scratchSizeInBytes*	
in	*overlapWindowSizeInPixels*	
in	*tileWidth*	
in	*tileHeight*	

4.14.2.10 OptixResult optixUtilDenoiserSplitImage (const OptixImage2D & *input*, const OptixImage2D & *output*, unsigned int *overlapWindowSizeInPixels*, unsigned int *tileWidth*, unsigned int *tileHeight*, std::vector< OptixUtilDenoiserImageTile > & *tiles*) `[inline]`

Split image into 2D tiles given horizontal and vertical tile size.

Parameters

in	*input*	full resolution input image to be split
in	*output*	full resolution output image
in	*overlapWindowSizeInPixels*	see OptixDenoiserSizes, optixDenoiserComputeMemoryResources
in	*tileWidth*	maximum width of tiles
in	*tileHeight*	maximum height of tiles
out	*tiles*	list of tiles covering the input image

4.14.2.11 OptixResult optixUtilGetPixelStride (const OptixImage2D & *image*, unsigned int & *pixelStrideInBytes*) `[inline]`

Return pixel stride in bytes for the given pixel format if the pixelStrideInBytes member of the image is zero. Otherwise return pixelStrideInBytes from the image.

Parameters

in	*image*	Image containing the pixel stride

5 Namespace Documentation

5.1 optix_impl Namespace Reference

Functions

- static __forceinline__
  __device__ void optixDumpStaticTransformFromHandle (OptixTraversableHandle handle)
- static __forceinline__
  __device__ void optixDumpMotionMatrixTransformFromHandle (OptixTraversableHandle handle)
- static __forceinline__
  __device__ void optixDumpSrtMatrixTransformFromHandle (OptixTraversableHandle handle)
- static __forceinline__
  __device__ void optixDumpInstanceFromHandle (OptixTraversableHandle handle)
- static __forceinline__
  __device__ void optixDumpTransform (OptixTraversableHandle handle)
- static __forceinline__
  __device__ void optixDumpTransformList ()
- static __forceinline__
  __device__ void optixDumpExceptionDetails ()
- static __forceinline__
  __device__ float4 optixAddFloat4 (const float4 &a, const float4 &b)
- static __forceinline__
  __device__ float4 optixMulFloat4 (const float4 &a, float b)
- static __forceinline__
  __device__ uint4 optixLdg (unsigned long long addr)
- template<class T >
  static __forceinline__ __device__ T optixLoadReadOnlyAlign16 (const T *ptr)
- static __forceinline__
  __device__ float4 optixMultiplyRowMatrix (const float4 vec, const float4 m0, const float4 m1, const float4 m2)
- static __forceinline__
  __device__ void optixGetMatrixFromSrt (float4 &m0, float4 &m1, float4 &m2, const OptixSRTData &srt)
- static __forceinline__
  __device__ void optixInvertMatrix (float4 &m0, float4 &m1, float4 &m2)
- static __forceinline__
  __device__ void optixLoadInterpolatedMatrixKey (float4 &m0, float4 &m1, float4 &m2, const float4 *matrix, const float t1)
- static __forceinline__
  __device__ void optixLoadInterpolatedSrtKey (float4 &srt0, float4 &srt1, float4 &srt2, float4 &srt3, const float4 *srt, const float t1)
- static __forceinline__
  __device__ void optixResolveMotionKey (float &localt, int &key, const OptixMotionOptions &options, const float globalt)

- static __forceinline__
  __device__ void optixGetInterpolatedTransformation (float4 &trf0, float4 &trf1, float4 &trf2, const OptixMatrixMotionTransform *transformData, const float time)
- static __forceinline__
  __device__ void optixGetInterpolatedTransformation (float4 &trf0, float4 &trf1, float4 &trf2, const OptixSRTMotionTransform *transformData, const float time)
- static __forceinline__
  __device__ void optixGetInterpolatedTransformationFromHandle (float4 &trf0, float4 &trf1, float4 &trf2, const OptixTraversableHandle handle, const float time, const bool objectToWorld)
- static __forceinline__
  __device__ void optixGetWorldToObjectTransformMatrix (float4 &m0, float4 &m1, float4 &m2)
- static __forceinline__
  __device__ void optixGetObjectToWorldTransformMatrix (float4 &m0, float4 &m1, float4 &m2)
- static __forceinline__
  __device__ float3 optixTransformPoint (const float4 &m0, const float4 &m1, const float4 &m2, const float3 &p)
- static __forceinline__
  __device__ float3 optixTransformVector (const float4 &m0, const float4 &m1, const float4 &m2, const float3 &v)
- static __forceinline__
  __device__ float3 optixTransformNormal (const float4 &m0, const float4 &m1, const float4 &m2, const float3 &n)

5.1.1 Function Documentation

5.1.1.1 static __forceinline__ __device__ float4 optix_impl::optixAddFloat4 (const float4 & *a*, const float4 & *b*) `[static]`

5.1.1.2 static __forceinline__ __device__ void optix_impl::optixDumpExceptionDetails ( ) `[static]`

5.1.1.3 static __forceinline__ __device__ void optix_impl::optixDumpInstanceFromHandle (OptixTraversableHandle *handle*) `[static]`

5.1.1.4 static __forceinline__ __device__ void optix_impl::optixDumpMotionMatrixTransformFromHandle (OptixTraversableHandle *handle*) `[static]`

5.1.1.5 static __forceinline__ __device__ void optix_impl::optixDumpSrtMatrixTransformFromHandle (OptixTraversableHandle *handle*) `[static]`

5.1.1.6 static __forceinline__ __device__ void optix_impl::optixDumpStaticTransformFromHandle (

OptixTraversableHandle *handle*) `[static]`

5.1.1.7 static __forceinline__ __device__ void optix_impl::optixDumpTransform (OptixTraversableHandle *handle*) `[static]`

5.1.1.8 static __forceinline__ __device__ void optix_impl::optixDumpTransformList ( ) `[static]`

5.1.1.9 static __forceinline__ __device__ void optix_impl::optixGetInterpolatedTransformation (

float4 & *trf0,*
float4 & *trf1,*
float4 & *trf2,*
const OptixMatrixMotionTransform ∗ *transformData,*
const float *time*) `[static]`

5.1.1.10 static __forceinline__ __device__ void optix_impl::optixGetInterpolatedTransformation (

float4 & *trf0,*
float4 & *trf1,*
float4 & *trf2,*
const OptixSRTMotionTransform ∗ *transformData,*
const float *time*) `[static]`

5.1.1.11 static __forceinline__ __device__ void optix_impl::optixGetInterpolatedTransformationFromHandle (

float4 & *trf0,*
float4 & *trf1,*
float4 & *trf2,*
const OptixTraversableHandle *handle,*
const float *time,*
const bool *objectToWorld*) `[static]`

5.1.1.12 static __forceinline__ __device__ void optix_impl::optixGetMatrixFromSrt (

float4 & *m0,*
float4 & *m1,*
float4 & *m2,*
const OptixSRTData & *srt*) `[static]`

5.1.1.13 static __forceinline__ __device__ void optix_impl::optixGetObjectToWorldTransformMatrix (

float4 & *m0,*
float4 & *m1,*

float4 & *m2*) `[static]`

5.1.1.14 static __forceinline__ __device__ void optix_impl::optixGetWorldToObjectTransformMatrix (

float4 & *m0,*

float4 & *m1,*

float4 & *m2*) `[static]`

5.1.1.15 static __forceinline__ __device__ void optix_impl::optixInvertMatrix (

float4 & *m0,*

float4 & *m1,*

float4 & *m2*) `[static]`

5.1.1.16 static __forceinline__ __device__ uint4 optix_impl::optixLdg (

unsigned long long *addr*) `[static]`

5.1.1.17 static __forceinline__ __device__ void optix_impl::optixLoadInterpolatedMatrixKey (

float4 & *m0,*

float4 & *m1,*

float4 & *m2,*

const float4 ∗ *matrix,*

const float *t1*) `[static]`

5.1.1.18 static __forceinline__ __device__ void optix_impl::optixLoadInterpolatedSrtKey (

float4 & *srt0,*

float4 & *srt1,*

float4 & *srt2,*

float4 & *srt3,*

const float4 ∗ *srt,*

const float *t1*) `[static]`

5.1.1.19 template<class T > static __forceinline__ __device__ T optix_impl::optixLoadReadOnlyAlign16 (

const T ∗ *ptr*) `[static]`

5.1.1.20 static __forceinline__ __device__ float4 optix_impl::optixMulFloat4 (

const float4 & *a,*

float *b*) `[static]`

5.1.1.21 static __forceinline__ __device__ float4 optix_impl::optixMultiplyRowMatrix (

const float4 *vec,*

const float4 *m0,*

const float4 *m1,*

const float4 *m2*) `[static]`

5.1.1.22 static __forceinline__ __device__ void optix_impl::optixResolveMotionKey (float & *localt*, int & *key*, const OptixMotionOptions & *options*, const float *globalt*) `[static]`

5.1.1.23 static __forceinline__ __device__ float3 optix_impl::optixTransformNormal (const float4 & *m0*, const float4 & *m1*, const float4 & *m2*, const float3 & *n*) `[static]`

5.1.1.24 static __forceinline__ __device__ float3 optix_impl::optixTransformPoint (const float4 & *m0*, const float4 & *m1*, const float4 & *m2*, const float3 & *p*) `[static]`

5.1.1.25 static __forceinline__ __device__ float3 optix_impl::optixTransformVector (const float4 & *m0*, const float4 & *m1*, const float4 & *m2*, const float3 & *v*) `[static]`

5.2 optix_internal Namespace Reference

Classes

- struct TypePack

6 Class Documentation

6.1 OptixAabb Struct Reference

Public Attributes

- float minX
- float minY
- float minZ
- float maxX
- float maxY
- float maxZ

6.1.1 Detailed Description

AABB inputs.

6.1.2 Member Data Documentation

6.1.2.1 float OptixAabb::maxX

Upper extent in X direction.

6.1.2.2 float OptixAabb::maxY

Upper extent in Y direction.

6.1.2.3 float OptixAabb::maxZ

Upper extent in Z direction.

6.1.2.4 float OptixAabb::minX

Lower extent in X direction.

6.1.2.5 float OptixAabb::minY

Lower extent in Y direction.

6.1.2.6 float OptixAabb::minZ

Lower extent in Z direction.

6.2 OptixAccelBufferSizes Struct Reference

Public Attributes

- size_t outputSizeInBytes
- size_t tempSizeInBytes
- size_t tempUpdateSizeInBytes

6.2.1 Detailed Description

Struct for querying builder allocation requirements.

Once queried the sizes should be used to allocate device memory of at least these sizes.

See Also

optixAccelComputeMemoryUsage()

6.2.2 Member Data Documentation

6.2.2.1 size_t OptixAccelBufferSizes::outputSizeInBytes

The size in bytes required for the outputBuffer parameter to optixAccelBuild when doing a build (OPTIX_BUILD_OPERATION_BUILD).

6.2.2.2 size_t OptixAccelBufferSizes::tempSizeInBytes

The size in bytes required for the tempBuffer paramter to optixAccelBuild when doing a build (OPTIX_BUILD_OPERATION_BUILD).

6.2.2.3 size_t OptixAccelBufferSizes::tempUpdateSizeInBytes

The size in bytes required for the tempBuffer parameter to optixAccelBuild when doing an update (OPTIX_BUILD_OPERATION_UPDATE). This value can be different than tempSizeInBytes used for a full build. Only non-zero if OPTIX_BUILD_FLAG_ALLOW_UPDATE flag is set in OptixAccelBuildOptions.

6.3 OptixAccelBuildOptions Struct Reference

Public Attributes

- unsigned int buildFlags
- OptixBuildOperation operation
- OptixMotionOptions motionOptions

6.3.1 Detailed Description

Build options for acceleration structures.

See Also

optixAccelComputeMemoryUsage(), optixAccelBuild()

6.3.2 Member Data Documentation

6.3.2.1 unsigned int OptixAccelBuildOptions::buildFlags

Combinations of OptixBuildFlags.

6.3.2.2 OptixMotionOptions OptixAccelBuildOptions::motionOptions

Options for motion.

6.3.2.3 OptixBuildOperation OptixAccelBuildOptions::operation

If OPTIX_BUILD_OPERATION_UPDATE the output buffer is assumed to contain the result of a full build with OPTIX_BUILD_FLAG_ALLOW_UPDATE set and using the same number of primitives. It is updated incrementally to reflect the current position of the primitives.

6.4 OptixAccelEmitDesc Struct Reference

Public Attributes

- CUdeviceptr result
- OptixAccelPropertyType type

6.4.1 Detailed Description

Specifies a type and output destination for emitted post-build properties.

See Also

optixAccelBuild()

6.4.2 Member Data Documentation

6.4.2.1 CUdeviceptr OptixAccelEmitDesc::result

Output buffer for the properties.

6.4.2.2 OptixAccelPropertyType OptixAccelEmitDesc::type

Requested property.

6.5 OptixAccelRelocationInfo Struct Reference

Public Attributes

- unsigned long long info [4]

6.5.1 Detailed Description

Used to store information related to relocation of acceleration structures.

See Also

optixAccelGetRelocationInfo(), optixAccelCheckRelocationCompatibility(), optixAccelRelocate()

6.5.2 Member Data Documentation

6.5.2.1 unsigned long long OptixAccelRelocationInfo::info[4]

Opaque data, used internally, should not be modified.

6.6 OptixBuildInput Struct Reference

Public Attributes

- OptixBuildInputType type
- union {
  OptixBuildInputTriangleArray triangleArray
  OptixBuildInputCurveArray curveArray
  OptixBuildInputSphereArray sphereArray
  OptixBuildInputCustomPrimitiveArray customPrimitiveArray
  OptixBuildInputInstanceArray instanceArray
  char pad [1024]
  };

6.6.1 Detailed Description

Build inputs.

All of them support motion and the size of the data arrays needs to match the number of motion steps

See Also

optixAccelComputeMemoryUsage(), optixAccelBuild()

6.6.2 Member Data Documentation

6.6.2.1 union { ... }

6.6.2.2 OptixBuildInputCurveArray OptixBuildInput::curveArray

Curve inputs.

6.6.2.3 OptixBuildInputCustomPrimitiveArray OptixBuildInput::customPrimitiveArray

Custom primitive inputs.

6.6.2.4 OptixBuildInputInstanceArray OptixBuildInput::instanceArray

Instance and instance pointer inputs.

6.6.2.5 char OptixBuildInput::pad[1024]

6.6.2.6 OptixBuildInputSphereArray OptixBuildInput::sphereArray

Sphere inputs.

6.6.2.7 OptixBuildInputTriangleArray OptixBuildInput::triangleArray

Triangle inputs.

6.6.2.8 OptixBuildInputType OptixBuildInput::type

The type of the build input.

6.7 OptixBuildInputCurveArray Struct Reference

Public Attributes

- OptixPrimitiveType curveType
- unsigned int numPrimitives
- const CUdeviceptr ∗ vertexBuffers
- unsigned int numVertices
- unsigned int vertexStrideInBytes
- const CUdeviceptr ∗ widthBuffers
- unsigned int widthStrideInBytes
- const CUdeviceptr ∗ normalBuffers
- unsigned int normalStrideInBytes
- CUdeviceptr indexBuffer
- unsigned int indexStrideInBytes
- unsigned int flag
- unsigned int primitiveIndexOffset
- unsigned int endcapFlags

6.7.1 Detailed Description

Curve inputs.

A curve is a swept surface defined by a 3D spline curve and a varying width (radius). A curve (or "strand") of degree d (3=cubic, 2=quadratic, 1=linear) is represented by N $>$ d vertices and N width values, and comprises N - d segments. Each segment is defined by d+1 consecutive vertices. Each curve may have a different number of vertices.

OptiX describes the curve array as a list of curve segments. The primitive id is the segment number. It is the user's responsibility to maintain a mapping between curves and curve segments. Each index buffer entry i = indexBuffer[primid] specifies the start of a curve segment, represented by d+1 consecutive vertices in the vertex buffer, and d+1 consecutive widths in the width buffer. Width is interpolated the same way vertices are interpolated, that is, using the curve basis.

Each curves build input has only one SBT record. To create curves with different materials in the same BVH, use multiple build inputs.

See Also

OptixBuildInput::curveArray

6.7.2 Member Data Documentation

6.7.2.1 OptixPrimitiveType OptixBuildInputCurveArray::curveType

Curve degree and basis.

See Also

OptixPrimitiveType

6.7.2.2 unsigned int OptixBuildInputCurveArray::endcapFlags

End cap flags, see OptixCurveEndcapFlags.

6.7.2.3 unsigned int OptixBuildInputCurveArray::flag

Combination of OptixGeometryFlags describing the primitive behavior.

6.7.2.4 CUdeviceptr OptixBuildInputCurveArray::indexBuffer

Device pointer to array of unsigned ints, one per curve segment. This buffer is required (unlike for OptixBuildInputTriangleArray). Each index is the start of degree+1 consecutive vertices in vertexBuffers, and corresponding widths in widthBuffers and normals in normalBuffers. These define a single segment. Size of array is numPrimitives.

6.7.2.5 unsigned int OptixBuildInputCurveArray::indexStrideInBytes

Stride between indices. If set to zero, indices are assumed to be tightly packed and stride is sizeof(unsigned int).

6.7.2.6 const CUdeviceptr* OptixBuildInputCurveArray::normalBuffers

Reserved for future use.

6.7.2.7 unsigned int OptixBuildInputCurveArray::normalStrideInBytes

Reserved for future use.

6.7.2.8 unsigned int OptixBuildInputCurveArray::numPrimitives

Number of primitives. Each primitive is a polynomial curve segment.

6.7.2.9 unsigned int OptixBuildInputCurveArray::numVertices

Number of vertices in each buffer in vertexBuffers.

6.7.2.10 unsigned int OptixBuildInputCurveArray::primitiveIndexOffset

Primitive index bias, applied in optixGetPrimitiveIndex(). Sum of primitiveIndexOffset and number of primitives must not overflow 32bits.

6.7.2.11 const CUdeviceptr∗ OptixBuildInputCurveArray::vertexBuffers

Pointer to host array of device pointers, one per motion step. Host array size must match number of motion keys as set in OptixMotionOptions (or an array of size 1 if OptixMotionOptions::numKeys is set to 1). Each per-motion-key device pointer must point to an array of floats (the vertices of the curves).

6.7.2.12 unsigned int OptixBuildInputCurveArray::vertexStrideInBytes

Stride between vertices. If set to zero, vertices are assumed to be tightly packed and stride is sizeof(float3).

6.7.2.13 const CUdeviceptr∗ OptixBuildInputCurveArray::widthBuffers

Parallel to vertexBuffers: a device pointer per motion step, each with numVertices float values, specifying the curve width (radius) corresponding to each vertex.

6.7.2.14 unsigned int OptixBuildInputCurveArray::widthStrideInBytes

Stride between widths. If set to zero, widths are assumed to be tightly packed and stride is sizeof(float).

6.8 OptixBuildInputCustomPrimitiveArray Struct Reference

Public Attributes

- const CUdeviceptr ∗ aabbBuffers
- unsigned int numPrimitives
- unsigned int strideInBytes
- const unsigned int ∗ flags
- unsigned int numSbtRecords
- CUdeviceptr sbtIndexOffsetBuffer
- unsigned int sbtIndexOffsetSizeInBytes
- unsigned int sbtIndexOffsetStrideInBytes
- unsigned int primitiveIndexOffset

6.8.1 Detailed Description

Custom primitive inputs.

See Also

OptixBuildInput::customPrimitiveArray

6.8.2 Member Data Documentation

6.8.2.1 const CUdeviceptr∗ OptixBuildInputCustomPrimitiveArray::aabbBuffers

Points to host array of device pointers to AABBs (type OptixAabb), one per motion step. Host array size must match number of motion keys as set in OptixMotionOptions (or an array of size 1 if OptixMotionOptions::numKeys is set to 1). Each device pointer must be a multiple of OPTIX_AABB_BUFFER_BYTE_ALIGNMENT.

6.8.2.2 const unsigned int∗ OptixBuildInputCustomPrimitiveArray::flags

Array of flags, to specify flags per sbt record, combinations of OptixGeometryFlags describing the primitive behavior, size must match numSbtRecords.

6.8.2.3 unsigned int OptixBuildInputCustomPrimitiveArray::numPrimitives

Number of primitives in each buffer (i.e., per motion step) in OptixBuildInputCustomPrimitiveArray::aabbBuffers.

6.8.2.4 unsigned int OptixBuildInputCustomPrimitiveArray::numSbtRecords

Number of sbt records available to the sbt index offset override.

6.8.2.5 unsigned int OptixBuildInputCustomPrimitiveArray::primitiveIndexOffset

Primitive index bias, applied in optixGetPrimitiveIndex(). Sum of primitiveIndexOffset and number of primitive must not overflow 32bits.

6.8.2.6 CUdeviceptr OptixBuildInputCustomPrimitiveArray::sbtIndexOffsetBuffer

Device pointer to per-primitive local sbt index offset buffer. May be NULL. Every entry must be in range [0,numSbtRecords-1]. Size needs to be the number of primitives.

6.8.2.7 unsigned int OptixBuildInputCustomPrimitiveArray::sbtIndexOffsetSizeInBytes

Size of type of the sbt index offset. Needs to be 0, 1, 2 or 4 (8, 16 or 32 bit).

6.8.2.8 unsigned int OptixBuildInputCustomPrimitiveArray::sbtIndexOffsetStrideInBytes

Stride between the index offsets. If set to zero, the offsets are assumed to be tightly packed and the stride matches the size of the type (sbtIndexOffsetSizeInBytes).

6.8.2.9 unsigned int OptixBuildInputCustomPrimitiveArray::strideInBytes

Stride between AABBs (per motion key). If set to zero, the aabbs are assumed to be tightly packed and the stride is assumed to be sizeof(OptixAabb). If non-zero, the value must be a multiple of OPTIX_AABB_BUFFER_BYTE_ALIGNMENT.

6.9 OptixBuildInputInstanceArray Struct Reference

Public Attributes

- CUdeviceptr instances
- unsigned int numInstances

6.9.1 Detailed Description

Instance and instance pointer inputs.

See Also

OptixBuildInput::instanceArray

6.9.2 Member Data Documentation

6.9.2.1 CUdeviceptr OptixBuildInputInstanceArray::instances

If OptixBuildInput::type is OPTIX_BUILD_INPUT_TYPE_INSTANCE_POINTERS instances and aabbs should be interpreted as arrays of pointers instead of arrays of structs.

This pointer must be a multiple of OPTIX_INSTANCE_BYTE_ALIGNMENT if OptixBuildInput::type is OPTIX_BUILD_INPUT_TYPE_INSTANCES. The array elements must be a multiple of OPTIX_INSTANCE_BYTE_ALIGNMENT if OptixBuildInput::type is OPTIX_BUILD_INPUT_TYPE_INSTANCE_POINTERS.

6.9.2.2 unsigned int OptixBuildInputInstanceArray::numInstances

Number of elements in OptixBuildInputInstanceArray::instances.

6.10 OptixBuildInputSphereArray Struct Reference

Public Attributes

- const CUdeviceptr ∗ vertexBuffers
- unsigned int vertexStrideInBytes
- unsigned int numVertices
- const CUdeviceptr ∗ radiusBuffers
- unsigned int radiusStrideInBytes
- int singleRadius
- const unsigned int ∗ flags
- unsigned int numSbtRecords
- CUdeviceptr sbtIndexOffsetBuffer
- unsigned int sbtIndexOffsetSizeInBytes
- unsigned int sbtIndexOffsetStrideInBytes
- unsigned int primitiveIndexOffset

6.10.1 Detailed Description

Sphere inputs.

A sphere is defined by a center point and a radius. Each center point is represented by a vertex in the vertex buffer. There is either a single radius for all spheres, or the radii are represented by entries in the radius buffer.

The vertex buffers and radius buffers point to a host array of device pointers, one per motion step. Host array size must match the number of motion keys as set in OptixMotionOptions (or an array of size 1 if OptixMotionOptions::numKeys is set to 0 or 1). Each per motion key device pointer must point to an array of vertices corresponding to the center points of the spheres, or an array of 1 or N radii. Format OPTIX_VERTEX_FORMAT_FLOAT3 is used for vertices, OPTIX_VERTEX_FORMAT_FLOAT for radii.

See Also

OptixBuildInput::sphereArray

6.10.2 Member Data Documentation

6.10.2.1 const unsigned int∗ OptixBuildInputSphereArray::flags

Array of flags, to specify flags per sbt record, combinations of OptixGeometryFlags describing the primitive behavior, size must match numSbtRecords.

6.10.2.2 unsigned int OptixBuildInputSphereArray::numSbtRecords

Number of sbt records available to the sbt index offset override.

6.10.2.3 unsigned int OptixBuildInputSphereArray::numVertices

Number of vertices in each buffer in vertexBuffers.

6.10.2.4 unsigned int OptixBuildInputSphereArray::primitiveIndexOffset

Primitive index bias, applied in optixGetPrimitiveIndex(). Sum of primitiveIndexOffset and number of primitives must not overflow 32bits.

6.10.2.5 const CUdeviceptr∗ OptixBuildInputSphereArray::radiusBuffers

Parallel to vertexBuffers: a device pointer per motion step, each with numRadii float values, specifying the sphere radius corresponding to each vertex.

6.10.2.6 unsigned int OptixBuildInputSphereArray::radiusStrideInBytes

Stride between radii. If set to zero, widths are assumed to be tightly packed and stride is sizeof(float).

6.10.2.7 CUdeviceptr OptixBuildInputSphereArray::sbtIndexOffsetBuffer

Device pointer to per-primitive local sbt index offset buffer. May be NULL. Every entry must be in range [0,numSbtRecords-1]. Size needs to be the number of primitives.

6.10.2.8 unsigned int OptixBuildInputSphereArray::sbtIndexOffsetSizeInBytes

Size of type of the sbt index offset. Needs to be 0, 1, 2 or 4 (8, 16 or 32 bit).

6.10.2.9 unsigned int OptixBuildInputSphereArray::sbtIndexOffsetStrideInBytes

Stride between the sbt index offsets. If set to zero, the offsets are assumed to be tightly packed and the stride matches the size of the type (sbtIndexOffsetSizeInBytes).

6.10.2.10 int OptixBuildInputSphereArray::singleRadius

Boolean value indicating whether a single radius per radius buffer is used, or the number of radii in radiusBuffers equals numVertices.

6.10.2.11 const CUdeviceptr* OptixBuildInputSphereArray::vertexBuffers

Pointer to host array of device pointers, one per motion step. Host array size must match number of motion keys as set in OptixMotionOptions (or an array of size 1 if OptixMotionOptions::numKeys is set to 1). Each per-motion-key device pointer must point to an array of floats (the center points of the spheres).

6.10.2.12 unsigned int OptixBuildInputSphereArray::vertexStrideInBytes

Stride between vertices. If set to zero, vertices are assumed to be tightly packed and stride is sizeof(float3).

6.11 OptixBuildInputTriangleArray Struct Reference

Public Attributes

- const CUdeviceptr * vertexBuffers
- unsigned int numVertices
- OptixVertexFormat vertexFormat
- unsigned int vertexStrideInBytes
- CUdeviceptr indexBuffer
- unsigned int numIndexTriplets
- OptixIndicesFormat indexFormat
- unsigned int indexStrideInBytes
- CUdeviceptr preTransform
- const unsigned int * flags
- unsigned int numSbtRecords
- CUdeviceptr sbtIndexOffsetBuffer
- unsigned int sbtIndexOffsetSizeInBytes
- unsigned int sbtIndexOffsetStrideInBytes
- unsigned int primitiveIndexOffset
- OptixTransformFormat transformFormat

6.11.1 Detailed Description

Triangle inputs.

See Also

OptixBuildInput::triangleArray

6.11.2 Member Data Documentation

6.11.2.1 const unsigned int∗ OptixBuildInputTriangleArray::flags

Array of flags, to specify flags per sbt record, combinations of OptixGeometryFlags describing the primitive behavior, size must match numSbtRecords.

6.11.2.2 CUdeviceptr OptixBuildInputTriangleArray::indexBuffer

Optional pointer to array of 16 or 32-bit int triplets, one triplet per triangle. The minimum alignment must match the natural alignment of the type as specified in the indexFormat, i.e., for OPTIX_INDICES_FORMAT_UNSIGNED_INT3 4-byte and for OPTIX_INDICES_FORMAT_UNSIGNED_SHORT3 a 2-byte alignment.

6.11.2.3 OptixIndicesFormat OptixBuildInputTriangleArray::indexFormat

See Also

OptixIndicesFormat

6.11.2.4 unsigned int OptixBuildInputTriangleArray::indexStrideInBytes

Stride between triplets of indices. If set to zero, indices are assumed to be tightly packed and stride is inferred from indexFormat.

6.11.2.5 unsigned int OptixBuildInputTriangleArray::numIndexTriplets

Size of array in OptixBuildInputTriangleArray::indexBuffer. For build, needs to be zero if indexBuffer is `nullptr`.

6.11.2.6 unsigned int OptixBuildInputTriangleArray::numSbtRecords

Number of sbt records available to the sbt index offset override.

6.11.2.7 unsigned int OptixBuildInputTriangleArray::numVertices

Number of vertices in each of buffer in OptixBuildInputTriangleArray::vertexBuffers.

6.11.2.8 CUdeviceptr OptixBuildInputTriangleArray::preTransform

Optional pointer to array of floats representing a 3x4 row major affine transformation matrix. This pointer must be a multiple of OPTIX_GEOMETRY_TRANSFORM_BYTE_ALIGNMENT.

6.11.2.9 unsigned int OptixBuildInputTriangleArray::primitiveIndexOffset

Primitive index bias, applied in optixGetPrimitiveIndex(). Sum of primitiveIndexOffset and number of triangles must not overflow 32bits.

6.11.2.10 CUdeviceptr OptixBuildInputTriangleArray::sbtIndexOffsetBuffer

Device pointer to per-primitive local sbt index offset buffer. May be NULL. Every entry must be in range [0,numSbtRecords-1]. Size needs to be the number of primitives.

6.11.2.11 unsigned int OptixBuildInputTriangleArray::sbtIndexOffsetSizeInBytes

Size of type of the sbt index offset. Needs to be 0, 1, 2 or 4 (8, 16 or 32 bit).

6.11.2.12 unsigned int OptixBuildInputTriangleArray::sbtIndexOffsetStrideInBytes

Stride between the index offsets. If set to zero, the offsets are assumed to be tightly packed and the stride matches the size of the type (sbtIndexOffsetSizeInBytes).

6.11.2.13 OptixTransformFormat OptixBuildInputTriangleArray::transformFormat

See Also

OptixTransformFormat

6.11.2.14 const CUdeviceptr∗ OptixBuildInputTriangleArray::vertexBuffers

Points to host array of device pointers, one per motion step. Host array size must match the number of motion keys as set in OptixMotionOptions (or an array of size 1 if OptixMotionOptions::numKeys is set to 0 or 1). Each per motion key device pointer must point to an array of vertices of the triangles in the format as described by vertexFormat. The minimum alignment must match the natural alignment of the type as specified in the vertexFormat, i.e., for OPTIX_VERTEX_FORMAT_FLOATX 4-byte, for all others a 2-byte alignment. However, an 16-byte stride (and buffer alignment) is recommended for vertices of format OPTIX_VERTEX_FORMAT_FLOAT3 for GAS build performance.

6.11.2.15 OptixVertexFormat OptixBuildInputTriangleArray::vertexFormat

See Also

OptixVertexFormat

6.11.2.16 unsigned int OptixBuildInputTriangleArray::vertexStrideInBytes

Stride between vertices. If set to zero, vertices are assumed to be tightly packed and stride is inferred from vertexFormat.

6.12 OptixBuiltinISOptions Struct Reference

Public Attributes

- OptixPrimitiveType builtinISModuleType

- int usesMotionBlur
- unsigned int buildFlags
- unsigned int curveEndcapFlags

6.12.1 Detailed Description

Specifies the options for retrieving an intersection program for a built-in primitive type. The primitive type must not be OPTIX_PRIMITIVE_TYPE_CUSTOM.

See Also

optixBuiltinISModuleGet()

6.12.2 Member Data Documentation

6.12.2.1 unsigned int OptixBuiltinISOptions::buildFlags

Build flags, see OptixBuildFlags.

6.12.2.2 OptixPrimitiveType OptixBuiltinISOptions::builtinISModuleType

6.12.2.3 unsigned int OptixBuiltinISOptions::curveEndcapFlags

End cap properties of curves, see OptixCurveEndcapFlags, 0 for non-curve types.

6.12.2.4 int OptixBuiltinISOptions::usesMotionBlur

Boolean value indicating whether vertex motion blur is used (but not motion transform blur).

6.13 OptixDenoiserGuideLayer Struct Reference

Public Attributes

- OptixImage2D albedo
- OptixImage2D normal
- OptixImage2D flow
- OptixImage2D previousOutputInternalGuideLayer
- OptixImage2D outputInternalGuideLayer

6.13.1 Detailed Description

Guide layer for the denoiser.

See Also

optixDenoiserInvoke()

6.13.2 Member Data Documentation

6.13.2.1 OptixImage2D OptixDenoiserGuideLayer::albedo

6.13.2.2 OptixImage2D OptixDenoiserGuideLayer::flow

6.13.2.3 OptixImage2D OptixDenoiserGuideLayer::normal

6.13.2.4 OptixImage2D OptixDenoiserGuideLayer::outputInternalGuideLayer

6.13.2.5 OptixImage2D OptixDenoiserGuideLayer::previousOutputInternalGuideLayer

6.14 OptixDenoiserLayer Struct Reference

Public Attributes

- OptixImage2D input
- OptixImage2D previousOutput
- OptixImage2D output

6.14.1 Detailed Description

Input/Output layers for the denoiser.

See Also

optixDenoiserInvoke()

6.14.2 Member Data Documentation

6.14.2.1 OptixImage2D OptixDenoiserLayer::input

6.14.2.2 OptixImage2D OptixDenoiserLayer::output

6.14.2.3 OptixImage2D OptixDenoiserLayer::previousOutput

6.15 OptixDenoiserOptions Struct Reference

Public Attributes

- unsigned int guideAlbedo
- unsigned int guideNormal

6.15.1 Detailed Description

Options used by the denoiser.

See Also

optixDenoiserCreate()

6.15.2 Member Data Documentation

6.15.2.1 unsigned int OptixDenoiserOptions::guideAlbedo

6.15.2.2 unsigned int OptixDenoiserOptions::guideNormal

6.16 OptixDenoiserParams Struct Reference

Public Attributes

- OptixDenoiserAlphaMode denoiseAlpha
- CUdeviceptr hdrIntensity
- float blendFactor
- CUdeviceptr hdrAverageColor
- unsigned int temporalModeUsePreviousLayers

6.16.1 Member Data Documentation

6.16.1.1 float OptixDenoiserParams::blendFactor

blend factor. If set to 0 the output is 100% of the denoised input. If set to 1, the output is 100% of the unmodified input. Values between 0 and 1 will linearly interpolate between the denoised and unmodified input.

6.16.1.2 OptixDenoiserAlphaMode OptixDenoiserParams::denoiseAlpha

alpha denoise mode

6.16.1.3 CUdeviceptr OptixDenoiserParams::hdrAverageColor

this parameter is used when the OPTIX_DENOISER_MODEL_KIND_AOV model kind is set. average log color of input image, separate for RGB channels (default null pointer). points to three floats. with the default (null pointer) denoised results will not be optimal.

6.16.1.4 CUdeviceptr OptixDenoiserParams::hdrIntensity

average log intensity of input image (default null pointer). points to a single float. with the default (null pointer) denoised results will not be optimal for very dark or bright input images.

6.16.1.5 unsigned int OptixDenoiserParams::temporalModeUsePreviousLayers

In temporal modes this parameter must be set to 1 if previous layers (e.g. previousOutputInternalGuideLayer) contain valid data. This is the case in the second and subsequent frames of a sequence (for example after a change of camera angle). In the first frame of such a sequence this parameter must be set to 0.

6.17 OptixDenoiserSizes Struct Reference

Public Attributes

- size_t stateSizeInBytes
- size_t withOverlapScratchSizeInBytes
- size_t withoutOverlapScratchSizeInBytes
- unsigned int overlapWindowSizeInPixels
- size_t computeAverageColorSizeInBytes
- size_t computeIntensitySizeInBytes
- size_t internalGuideLayerPixelSizeInBytes

6.17.1 Detailed Description

Various sizes related to the denoiser.

See Also

optixDenoiserComputeMemoryResources()

6.17.2 Member Data Documentation

6.17.2.1 size_t OptixDenoiserSizes::computeAverageColorSizeInBytes

Size of scratch memory passed to optixDenoiserComputeAverageColor. The size is independent of the tile/image resolution.

6.17.2.2 size_t OptixDenoiserSizes::computeIntensitySizeInBytes

Size of scratch memory passed to optixDenoiserComputeIntensity. The size is independent of the tile/image resolution.

6.17.2.3 size_t OptixDenoiserSizes::internalGuideLayerPixelSizeInBytes

Number of bytes for each pixel in internal guide layers.

6.17.2.4 unsigned int OptixDenoiserSizes::overlapWindowSizeInPixels

Overlap on all four tile sides.

6.17.2.5 size_t OptixDenoiserSizes::stateSizeInBytes

Size of state memory passed to optixDenoiserSetup, optixDenoiserInvoke.

6.17.2.6 size_t OptixDenoiserSizes::withoutOverlapScratchSizeInBytes

Size of scratch memory passed to optixDenoiserSetup, optixDenoiserInvoke. No overlap added.

6.17.2.7 size_t OptixDenoiserSizes::withOverlapScratchSizeInBytes

Size of scratch memory passed to optixDenoiserSetup, optixDenoiserInvoke. Overlap added to dimensions passed to optixDenoiserComputeMemoryResources.

6.18 OptixDeviceContextOptions Struct Reference

Public Attributes

- OptixLogCallback logCallbackFunction
- void ∗ logCallbackData
- int logCallbackLevel
- OptixDeviceContextValidationMode validationMode

6.18.1 Detailed Description

Parameters used for optixDeviceContextCreate()

See Also

optixDeviceContextCreate()

6.18.2 Member Data Documentation

6.18.2.1 void∗ OptixDeviceContextOptions::logCallbackData

Pointer stored and passed to logCallbackFunction when a message is generated.

6.18.2.2 OptixLogCallback OptixDeviceContextOptions::logCallbackFunction

Function pointer used when OptiX wishes to generate messages.

6.18.2.3 int OptixDeviceContextOptions::logCallbackLevel

Maximum callback level to generate message for (see OptixLogCallback)

6.18.2.4 OptixDeviceContextValidationMode OptixDeviceContextOptions::validationMode

Validation mode of context.

6.19 OptixFunctionTable Struct Reference

Public Attributes

Error handling

- const char ∗(∗ optixGetErrorName)(OptixResult result)
- const char ∗(∗ optixGetErrorString)(OptixResult result)

Device context

- OptixResult(∗ optixDeviceContextCreate)(CUcontext fromContext, const OptixDeviceContextOptions ∗options, OptixDeviceContext ∗context)
- OptixResult(∗ optixDeviceContextDestroy)(OptixDeviceContext context)
- OptixResult(∗ optixDeviceContextGetProperty)(OptixDeviceContext context, OptixDeviceProperty property, void ∗value, size_t sizeInBytes)
- OptixResult(∗ optixDeviceContextSetLogCallback)(OptixDeviceContext context, OptixLogCallback callbackFunction, void ∗callbackData, unsigned int callbackLevel)
- OptixResult(∗ optixDeviceContextSetCacheEnabled)(OptixDeviceContext context, int enabled)
- OptixResult(∗ optixDeviceContextSetCacheLocation)(OptixDeviceContext context, const char ∗location)
- OptixResult(∗ optixDeviceContextSetCacheDatabaseSizes)(OptixDeviceContext context, size_t lowWaterMark, size_t highWaterMark)
- OptixResult(∗ optixDeviceContextGetCacheEnabled)(OptixDeviceContext context, int ∗enabled)
- OptixResult(∗ optixDeviceContextGetCacheLocation)(OptixDeviceContext context, char ∗location, size_t locationSize)
- OptixResult(∗ optixDeviceContextGetCacheDatabaseSizes)(OptixDeviceContext context, size_t ∗lowWaterMark, size_t ∗highWaterMark)

Modules

- OptixResult(∗ optixModuleCreateFromPTX)(OptixDeviceContext context, const OptixModuleCompileOptions ∗moduleCompileOptions, const OptixPipelineCompileOptions ∗pipelineCompileOptions, const char ∗PTX, size_t PTXsize, char ∗logString, size_t ∗logStringSize, OptixModule ∗module)
- OptixResult(∗ optixModuleCreateFromPTXWithTasks)(OptixDeviceContext context, const OptixModuleCompileOptions ∗moduleCompileOptions, const OptixPipelineCompileOptions ∗pipelineCompileOptions, const char ∗PTX, size_t PTXsize, char ∗logString, size_t ∗logStringSize, OptixModule ∗module, OptixTask ∗firstTask)
- OptixResult(∗ optixModuleGetCompilationState)(OptixModule module, OptixModuleCompileState ∗state)
- OptixResult(∗ optixModuleDestroy)(OptixModule module)
- OptixResult(∗ optixBuiltinISModuleGet)(OptixDeviceContext context, const OptixModuleCompileOptions ∗moduleCompileOptions, const OptixPipelineCompileOptions ∗pipelineCompileOptions, const OptixBuiltinISOptions ∗builtinISOptions, OptixModule ∗builtinModule)

Tasks

- OptixResult(∗ optixTaskExecute)(OptixTask task, OptixTask ∗additionalTasks, unsigned int maxNumAdditionalTasks, unsigned int ∗numAdditionalTasksCreated)

Program groups

- OptixResult(∗ optixProgramGroupCreate)(OptixDeviceContext context, const OptixProgramGroupDesc ∗programDescriptions, unsigned int numProgramGroups, const OptixProgramGroupOptions ∗options, char ∗logString, size_t ∗logStringSize, OptixProgramGroup ∗programGroups)
- OptixResult(∗ optixProgramGroupDestroy)(OptixProgramGroup programGroup)

- OptixResult(∗ optixProgramGroupGetStackSize)(OptixProgramGroup programGroup, OptixStackSizes ∗stackSizes)

Pipeline

- OptixResult(∗ optixPipelineCreate)(OptixDeviceContext context, const OptixPipelineCompileOptions ∗pipelineCompileOptions, const OptixPipelineLinkOptions ∗pipelineLinkOptions, const OptixProgramGroup ∗programGroups, unsigned int numProgramGroups, char ∗logString, size_t ∗logStringSize, OptixPipeline ∗pipeline)
- OptixResult(∗ optixPipelineDestroy)(OptixPipeline pipeline)
- OptixResult(∗ optixPipelineSetStackSize)(OptixPipeline pipeline, unsigned int directCallableStackSizeFromTraversal, unsigned int directCallableStackSizeFromState, unsigned int continuationStackSize, unsigned int maxTraversableGraphDepth)

Acceleration structures

- OptixResult(∗ optixAccelComputeMemoryUsage)(OptixDeviceContext context, const OptixAccelBuildOptions ∗accelOptions, const OptixBuildInput ∗buildInputs, unsigned int numBuildInputs, OptixAccelBufferSizes ∗bufferSizes)
- OptixResult(∗ optixAccelBuild)(OptixDeviceContext context, CUstream stream, const OptixAccelBuildOptions ∗accelOptions, const OptixBuildInput ∗buildInputs, unsigned int numBuildInputs, CUdeviceptr tempBuffer, size_t tempBufferSizeInBytes, CUdeviceptr outputBuffer, size_t outputBufferSizeInBytes, OptixTraversableHandle ∗outputHandle, const OptixAccelEmitDesc ∗emittedProperties, unsigned int numEmittedProperties)
- OptixResult(∗ optixAccelGetRelocationInfo)(OptixDeviceContext context, OptixTraversableHandle handle, OptixAccelRelocationInfo ∗info)
- OptixResult(∗ optixAccelCheckRelocationCompatibility)(OptixDeviceContext context, const OptixAccelRelocationInfo ∗info, int ∗compatible)
- OptixResult(∗ optixAccelRelocate)(OptixDeviceContext context, CUstream stream, const OptixAccelRelocationInfo ∗info, CUdeviceptr instanceTraversableHandles, size_t numInstanceTraversableHandles, CUdeviceptr targetAccel, size_t targetAccelSizeInBytes, OptixTraversableHandle ∗targetHandle)
- OptixResult(∗ optixAccelCompact)(OptixDeviceContext context, CUstream stream, OptixTraversableHandle inputHandle, CUdeviceptr outputBuffer, size_t outputBufferSizeInBytes, OptixTraversableHandle ∗outputHandle)
- OptixResult(∗ optixConvertPointerToTraversableHandle)(OptixDeviceContext onDevice, CUdeviceptr pointer, OptixTraversableType traversableType, OptixTraversableHandle ∗traversableHandle)
- void(∗ reserved1)(void)
- void(∗ reserved2)(void)
- void(∗ reserved3)(void)
- void(∗ reserved4)(void)

Launch

- OptixResult(∗ optixSbtRecordPackHeader)(OptixProgramGroup programGroup, void ∗sbtRecordHeaderHostPointer)
- OptixResult(∗ optixLaunch)(OptixPipeline pipeline, CUstream stream, CUdeviceptr pipelineParams, size_t pipelineParamsSize, const OptixShaderBindingTable ∗sbt, unsigned int width, unsigned int height, unsigned int depth)

Denoiser

- OptixResult(∗ optixDenoiserCreate)(OptixDeviceContext context, OptixDenoiserModelKind modelKind, const OptixDenoiserOptions ∗options, OptixDenoiser ∗returnHandle)
- OptixResult(∗ optixDenoiserDestroy)(OptixDenoiser handle)
- OptixResult(∗ optixDenoiserComputeMemoryResources)(const OptixDenoiser handle, unsigned int maximumInputWidth, unsigned int maximumInputHeight, OptixDenoiserSizes ∗returnSizes)
- OptixResult(∗ optixDenoiserSetup)(OptixDenoiser denoiser, CUstream stream, unsigned int inputWidth, unsigned int inputHeight, CUdeviceptr state, size_t stateSizeInBytes, CUdeviceptr scratch, size_t scratchSizeInBytes)
- OptixResult(∗ optixDenoiserInvoke)(OptixDenoiser denoiser, CUstream stream, const OptixDenoiserParams ∗params, CUdeviceptr denoiserState, size_t denoiserStateSizeInBytes, const OptixDenoiserGuideLayer ∗guideLayer, const OptixDenoiserLayer ∗layers, unsigned int numLayers, unsigned int inputOffsetX, unsigned int inputOffsetY, CUdeviceptr scratch, size_t scratchSizeInBytes)
- OptixResult(∗ optixDenoiserComputeIntensity)(OptixDenoiser handle, CUstream stream, const OptixImage2D ∗inputImage, CUdeviceptr outputIntensity, CUdeviceptr scratch, size_t scratchSizeInBytes)
- OptixResult(∗ optixDenoiserComputeAverageColor)(OptixDenoiser handle, CUstream stream, const OptixImage2D ∗inputImage, CUdeviceptr outputAverageColor, CUdeviceptr scratch, size_t scratchSizeInBytes)
- OptixResult(∗ optixDenoiserCreateWithUserModel)(OptixDeviceContext context, const void ∗data, size_t dataSizeInBytes, OptixDenoiser ∗returnHandle)

6.19.1 Detailed Description

The function table containing all API functions.

See optixInit() and optixInitWithHandle().

6.19.2 Member Data Documentation

6.19.2.1 OptixResult(∗ OptixFunctionTable::optixAccelBuild)(OptixDeviceContext context, CUstream stream, const OptixAccelBuildOptions ∗accelOptions, const OptixBuildInput ∗buildInputs, unsigned int numBuildInputs, CUdeviceptr tempBuffer, size_t tempBufferSizeInBytes, CUdeviceptr outputBuffer, size_t outputBufferSizeInBytes, OptixTraversableHandle ∗outputHandle, const OptixAccelEmitDesc ∗emittedProperties, unsigned int numEmittedProperties)

See optixAccelBuild().

6.19.2.2 OptixResult(∗ OptixFunctionTable::optixAccelCheckRelocationCompatibility)(OptixDeviceContext context, const OptixAccelRelocationInfo ∗info, int ∗compatible)

See optixAccelCheckRelocationCompatibility().

6.19.2.3 OptixResult(∗ OptixFunctionTable::optixAccelCompact)(OptixDeviceContext context, CUstream stream, OptixTraversableHandle inputHandle, CUdeviceptr outputBuffer, size_t outputBufferSizeInBytes, OptixTraversableHandle ∗outputHandle)

See optixAccelCompact().

6.19.2.4 OptixResult(∗ OptixFunctionTable::optixAccelComputeMemoryUsage)(OptixDeviceContext context, const OptixAccelBuildOptions ∗accelOptions, const OptixBuildInput ∗buildInputs, unsigned int numBuildInputs, OptixAccelBufferSizes ∗bufferSizes)

See optixAccelComputeMemoryUsage().

6.19.2.5 OptixResult(∗ OptixFunctionTable::optixAccelGetRelocationInfo)(OptixDeviceContext context, OptixTraversableHandle handle, OptixAccelRelocationInfo ∗info)

See optixAccelGetRelocationInfo().

6.19.2.6 OptixResult(∗ OptixFunctionTable::optixAccelRelocate)(OptixDeviceContext context, CUstream stream, const OptixAccelRelocationInfo ∗info, CUdeviceptr instanceTraversableHandles, size_t numInstanceTraversableHandles, CUdeviceptr targetAccel, size_t targetAccelSizeInBytes, OptixTraversableHandle ∗targetHandle)

See optixAccelRelocate().

6.19.2.7 OptixResult(∗ OptixFunctionTable::optixBuiltinISModuleGet)(OptixDeviceContext context, const OptixModuleCompileOptions ∗moduleCompileOptions, const OptixPipelineCompileOptions ∗pipelineCompileOptions, const OptixBuiltinISOptions ∗builtinISOptions, OptixModule ∗builtinModule)

See optixBuiltinISModuleGet().

6.19.2.8 OptixResult(∗ OptixFunctionTable::optixConvertPointerToTraversableHandle)(OptixDeviceContext onDevice, CUdeviceptr pointer, OptixTraversableType traversableType, OptixTraversableHandle ∗traversableHandle)

See optixConvertPointerToTraversableHandle().

6.19.2.9 OptixResult(∗ OptixFunctionTable::optixDenoiserComputeAverageColor)(OptixDenoiser handle, CUstream stream, const OptixImage2D ∗inputImage, CUdeviceptr outputAverageColor, CUdeviceptr scratch, size_t scratchSizeInBytes)

See optixDenoiserComputeAverageColor().

6.19.2.10 OptixResult(∗ OptixFunctionTable::optixDenoiserComputeIntensity)(OptixDenoiser handle, CUstream stream, const OptixImage2D ∗inputImage, CUdeviceptr outputIntensity, CUdeviceptr scratch, size_t scratchSizeInBytes)

See optixDenoiserComputeIntensity().

6.19.2.11 OptixResult(∗ OptixFunctionTable::optixDenoiserComputeMemoryResources)(const OptixDenoiser handle, unsigned int maximumInputWidth, unsigned int maximumInputHeight, OptixDenoiserSizes ∗returnSizes)

See optixDenoiserComputeMemoryResources().

6.19.2.12 OptixResult(∗ OptixFunctionTable::optixDenoiserCreate)(OptixDeviceContext context, OptixDenoiserModelKind modelKind, const OptixDenoiserOptions ∗options, OptixDenoiser ∗returnHandle)

See optixDenoiserCreate().

6.19.2.13 OptixResult(∗ OptixFunctionTable::optixDenoiserCreateWithUserModel)(OptixDeviceContext context, const void ∗data, size_t dataSizeInBytes, OptixDenoiser ∗returnHandle)

See optixDenoiserCreateWithUserModel().

6.19.2.14 OptixResult(∗ OptixFunctionTable::optixDenoiserDestroy)(OptixDenoiser handle)

See optixDenoiserDestroy().

6.19.2.15 OptixResult(∗ OptixFunctionTable::optixDenoiserInvoke)(OptixDenoiser denoiser, CUstream stream, const OptixDenoiserParams ∗params, CUdeviceptr denoiserState, size_t denoiserStateSizeInBytes, const OptixDenoiserGuideLayer ∗guideLayer, const OptixDenoiserLayer ∗layers, unsigned int numLayers, unsigned int inputOffsetX, unsigned int inputOffsetY, CUdeviceptr scratch, size_t scratchSizeInBytes)

See optixDenoiserInvoke().

6.19.2.16 OptixResult(∗ OptixFunctionTable::optixDenoiserSetup)(OptixDenoiser denoiser, CUstream stream, unsigned int inputWidth, unsigned int inputHeight, CUdeviceptr state, size_t stateSizeInBytes, CUdeviceptr scratch, size_t scratchSizeInBytes)

See optixDenoiserSetup().

6.19.2.17 OptixResult(∗ OptixFunctionTable::optixDeviceContextCreate)(CUcontext fromContext, const OptixDeviceContextOptions ∗options, OptixDeviceContext ∗context)

See optixDeviceContextCreate().

6.19.2.18 OptixResult(∗ OptixFunctionTable::optixDeviceContextDestroy)(OptixDeviceContext context)

See optixDeviceContextDestroy().

6.19.2.19 OptixResult(∗ OptixFunctionTable::optixDeviceContextGetCacheDatabaseSizes)(OptixDeviceContext context, size_t ∗lowWaterMark, size_t ∗highWaterMark)

See optixDeviceContextGetCacheDatabaseSizes().

6.19.2.20 OptixResult(∗ OptixFunctionTable::optixDeviceContextGetCacheEnabled)(OptixDeviceContext context, int ∗enabled)

See optixDeviceContextGetCacheEnabled().

6.19.2.21 OptixResult(∗ OptixFunctionTable::optixDeviceContextGetCacheLocation)(OptixDeviceContext context, char ∗location, size_t locationSize)

See optixDeviceContextGetCacheLocation().

6.19.2.22 OptixResult(∗ OptixFunctionTable::optixDeviceContextGetProperty)(OptixDeviceContext context, OptixDeviceProperty property, void ∗value, size_t sizeInBytes)

See optixDeviceContextGetProperty().

6.19.2.23 OptixResult(∗ OptixFunctionTable::optixDeviceContextSetCacheDatabaseSizes)(OptixDeviceContext context, size_t lowWaterMark, size_t highWaterMark)

See optixDeviceContextSetCacheDatabaseSizes().

6.19.2.24 OptixResult(∗ OptixFunctionTable::optixDeviceContextSetCacheEnabled)(OptixDeviceContext context, int enabled)

See optixDeviceContextSetCacheEnabled().

6.19.2.25 OptixResult(∗ OptixFunctionTable::optixDeviceContextSetCacheLocation)(OptixDeviceContext context, const char ∗location)

See optixDeviceContextSetCacheLocation().

6.19.2.26 OptixResult(∗ OptixFunctionTable::optixDeviceContextSetLogCallback)(OptixDeviceContext context, OptixLogCallback callbackFunction, void ∗callbackData, unsigned int callbackLevel)

See optixDeviceContextSetLogCallback().

6.19.2.27 const char∗(∗ OptixFunctionTable::optixGetErrorName)(OptixResult result)

See optixGetErrorName().

6.19.2.28 const char∗(∗ OptixFunctionTable::optixGetErrorString)(OptixResult result)

See optixGetErrorString().

6.19.2.29 OptixResult(∗ OptixFunctionTable::optixLaunch)(OptixPipeline pipeline, CUstream stream, CUdeviceptr pipelineParams, size_t pipelineParamsSize, const OptixShaderBindingTable ∗sbt, unsigned int width, unsigned int height, unsigned int depth)

See optixConvertPointerToTraversableHandle().

6.19.2.30 OptixResult(∗ OptixFunctionTable::optixModuleCreateFromPTX)(OptixDeviceContext context, const OptixModuleCompileOptions ∗moduleCompileOptions, const OptixPipelineCompileOptions ∗pipelineCompileOptions, const char ∗PTX, size_t PTXsize, char ∗logString, size_t ∗logStringSize, OptixModule ∗module)

See optixModuleCreateFromPTX().

6.19.2.31 OptixResult(∗ OptixFunctionTable::optixModuleCreateFromPTXWithTasks)(OptixDeviceContext context, const OptixModuleCompileOptions ∗moduleCompileOptions, const OptixPipelineCompileOptions ∗pipelineCompileOptions, const char ∗PTX, size_t PTXsize, char ∗logString, size_t ∗logStringSize, OptixModule ∗module, OptixTask ∗firstTask)

See optixModuleCreateFromPTXWithTasks().

6.19.2.32 OptixResult(∗ OptixFunctionTable::optixModuleDestroy)(OptixModule module)

See optixModuleDestroy().

6.19.2.33 OptixResult(∗ OptixFunctionTable::optixModuleGetCompilationState)(OptixModule module, OptixModuleCompileState ∗state)

See optixModuleGetCompilationState().

6.19.2.34 OptixResult(∗ OptixFunctionTable::optixPipelineCreate)(OptixDeviceContext context, const OptixPipelineCompileOptions ∗pipelineCompileOptions, const OptixPipelineLinkOptions ∗pipelineLinkOptions, const OptixProgramGroup ∗programGroups, unsigned int numProgramGroups, char ∗logString, size_t ∗logStringSize, OptixPipeline ∗pipeline)

See optixPipelineCreate().

6.19.2.35 OptixResult(∗ OptixFunctionTable::optixPipelineDestroy)(OptixPipeline pipeline)

See optixPipelineDestroy().

6.19.2.36 OptixResult(∗ OptixFunctionTable::optixPipelineSetStackSize)(OptixPipeline pipeline, unsigned int directCallableStackSizeFromTraversal, unsigned int directCallableStackSizeFromState, unsigned int continuationStackSize, unsigned int maxTraversableGraphDepth)

See optixPipelineSetStackSize().

6.19.2.37 OptixResult(∗ OptixFunctionTable::optixProgramGroupCreate)(OptixDeviceContext context, const OptixProgramGroupDesc ∗programDescriptions, unsigned int numProgramGroups, const OptixProgramGroupOptions ∗options, char ∗logString, size_t ∗logStringSize, OptixProgramGroup ∗programGroups)

See optixProgramGroupCreate().

6.19.2.38 OptixResult(∗ OptixFunctionTable::optixProgramGroupDestroy)(OptixProgramGroup programGroup)

See optixProgramGroupDestroy().

6.19.2.39 OptixResult(∗ OptixFunctionTable::optixProgramGroupGetStackSize)(OptixProgramGroup programGroup, OptixStackSizes ∗stackSizes)

See optixProgramGroupGetStackSize().

6.19.2.40 OptixResult(∗ OptixFunctionTable::optixSbtRecordPackHeader)(OptixProgramGroup programGroup, void ∗sbtRecordHeaderHostPointer)

See optixConvertPointerToTraversableHandle().

6.19.2.41 OptixResult(∗ OptixFunctionTable::optixTaskExecute)(OptixTask task, OptixTask ∗additionalTasks, unsigned int maxNumAdditionalTasks, unsigned int ∗numAdditionalTasksCreated)

See optixTaskExecute().

6.19.2.42 void(* OptixFunctionTable::reserved1)(void)

See optixAccelComputeMemoryUsage().

6.19.2.43 void(* OptixFunctionTable::reserved2)(void)

See optixAccelComputeMemoryUsage().

6.19.2.44 void(* OptixFunctionTable::reserved3)(void)

See optixAccelComputeMemoryUsage().

6.19.2.45 void(* OptixFunctionTable::reserved4)(void)

See optixAccelComputeMemoryUsage().

6.20 OptixImage2D Struct Reference

Public Attributes

- CUdeviceptr data
- unsigned int width
- unsigned int height
- unsigned int rowStrideInBytes
- unsigned int pixelStrideInBytes
- OptixPixelFormat format

6.20.1 Detailed Description

Image descriptor used by the denoiser.

See Also

optixDenoiserInvoke(), optixDenoiserComputeIntensity()

6.20.2 Member Data Documentation

6.20.2.1 CUdeviceptr OptixImage2D::data

Pointer to the actual pixel data.

6.20.2.2 OptixPixelFormat OptixImage2D::format

Pixel format.

6.20.2.3 unsigned int OptixImage2D::height

Height of the image (in pixels)

6.20.2.4 unsigned int OptixImage2D::pixelStrideInBytes

Stride between subsequent pixels of the image (in bytes). If set to 0, dense packing (no gaps) is assumed. For pixel format OPTIX_PIXEL_FORMAT_INTERNAL_GUIDE_LAYER it must be set to at least OptixDenoiserSizes::internalGuideLayerSizeInBytes.

6.20.2.5 unsigned int OptixImage2D::rowStrideInBytes

Stride between subsequent rows of the image (in bytes).

6.20.2.6 unsigned int OptixImage2D::width

Width of the image (in pixels)

6.21 OptixInstance Struct Reference

Public Attributes

- float transform [12]
- unsigned int instanceId
- unsigned int sbtOffset
- unsigned int visibilityMask
- unsigned int flags
- OptixTraversableHandle traversableHandle
- unsigned int pad [2]

6.21.1 Detailed Description

Instances.

See Also

OptixBuildInputInstanceArray::instances

6.21.2 Member Data Documentation

6.21.2.1 unsigned int OptixInstance::flags

Any combination of OptixInstanceFlags is allowed.

6.21.2.2 unsigned int OptixInstance::instanceId

Application supplied ID. The maximal ID can be queried using OPTIX_DEVICE_PROPERTY_LIMIT_MAX_INSTANCE_ID.

6.21.2.3 unsigned int OptixInstance::pad[2]

round up to 80-byte, to ensure 16-byte alignment

6.21.2.4 unsigned int OptixInstance::sbtOffset

SBT record offset. Will only be used for instances of geometry acceleration structure (GAS) objects. Needs to be set to 0 for instances of instance acceleration structure (IAS) objects. The maximal SBT offset can be queried using OPTIX_DEVICE_PROPERTY_LIMIT_MAX_INSTANCE_SBT_OFFSET.

6.21.2.5 float OptixInstance::transform[12]

affine object-to-world transformation as 3x4 matrix in row-major layout

6.21.2.6 OptixTraversableHandle OptixInstance::traversableHandle

Set with an OptixTraversableHandle.

6.21.2.7 unsigned int OptixInstance::visibilityMask

Visibility mask. If rayMask & instanceMask == 0 the instance is culled. The number of available bits can be queried using OPTIX_DEVICE_PROPERTY_LIMIT_NUM_BITS_INSTANCE_VISIBILITY_MASK.

6.22 OptixMatrixMotionTransform Struct Reference

Public Attributes

- OptixTraversableHandle child
- OptixMotionOptions motionOptions
- unsigned int pad [3]
- float transform [2][12]

6.22.1 Detailed Description

Represents a matrix motion transformation.

The device address of instances of this type must be a multiple of OPTIX_TRANSFORM_BYTE_ALIGNMENT.

This struct, as defined here, handles only N=2 motion keys due to the fixed array length of its transform member. The following example shows how to create instances for an arbitrary number N of motion keys:

```
float matrixData[N][12];
... // setup matrixData

size_t transformSizeInBytes = sizeof( OptixMatrixMotionTransform ) + ( N-2 ) * 12
       * sizeof( float );
OptixMatrixMotionTransform* matrixMoptionTransform = (
      OptixMatrixMotionTransform*) malloc( transformSizeInBytes );
memset( matrixMoptionTransform, 0, transformSizeInBytes );
```

```
... // setup other members of matrixMoptionTransform
matrixMoptionTransform->motionOptions.numKeys
memcpy( matrixMoptionTransform->transform, matrixData, N * 12 * sizeof( float ) );

... // copy matrixMoptionTransform to device memory
free( matrixMoptionTransform )
```

See Also

optixConvertPointerToTraversableHandle()

6.22.2 Member Data Documentation

6.22.2.1 OptixTraversableHandle OptixMatrixMotionTransform::child

The traversable that is transformed by this transformation.

6.22.2.2 OptixMotionOptions OptixMatrixMotionTransform::motionOptions

The motion options for this transformation.

6.22.2.3 unsigned int OptixMatrixMotionTransform::pad[3]

Padding to make the transformation 16 byte aligned.

6.22.2.4 float OptixMatrixMotionTransform::transform[2][12]

Affine object-to-world transformation as 3x4 matrix in row-major layout.

6.23 OptixModuleCompileBoundValueEntry Struct Reference

Public Attributes

- size_t pipelineParamOffsetInBytes
- size_t sizeInBytes
- const void ∗ boundValuePtr
- const char ∗ annotation

6.23.1 Detailed Description

Struct for specifying specializations for pipelineParams as specified in OptixPipelineCompileOptions::pipelineLaunchParamsVariableName.

The bound values are supposed to represent a constant value in the pipelineParams. OptiX will attempt to locate all loads from the pipelineParams and correlate them to the appropriate bound value, but there are cases where OptiX cannot safely or reliably do this. For example if the pointer to the pipelineParams is passed as an argument to a non-inline function or the offset of the load to the pipelineParams cannot be statically determined (e.g. accessed in a loop). No module should rely on the value being specialized in order to work correctly. The values in the pipelineParams specified on optixLaunch should match the bound value. If validation mode is enabled on the context, OptiX will verify that the bound values specified matches the values in pipelineParams specified to optixLaunch.

These values are compiled in to the module as constants. Once the constants are inserted into the code, an optimization pass will be run that will attempt to propagate the consants and remove unreachable code.

If caching is enabled, changes in these values will result in newly compiled modules.

The pipelineParamOffset and sizeInBytes must be within the bounds of the pipelineParams variable. OPTIX_ERROR_INVALID_VALUE will be returned from optixModuleCreateFromPTX otherwise.

If more than one bound value overlaps or the size of a bound value is equal to 0, an OPTIX_ERROR_INVALID_VALUE will be returned from optixModuleCreateFromPTX.

The same set of bound values do not need to be used for all modules in a pipeline, but overlapping values between modules must have the same value. OPTIX_ERROR_INVALID_VALUE will be returned from optixPipelineCreate otherwise.

See Also

OptixModuleCompileOptions

6.23.2 Member Data Documentation

6.23.2.1 const char∗ OptixModuleCompileBoundValueEntry::annotation

6.23.2.2 const void∗ OptixModuleCompileBoundValueEntry::boundValuePtr

6.23.2.3 size_t OptixModuleCompileBoundValueEntry::pipelineParamOffsetInBytes

6.23.2.4 size_t OptixModuleCompileBoundValueEntry::sizeInBytes

6.24 OptixModuleCompileOptions Struct Reference

Public Attributes

- int maxRegisterCount
- OptixCompileOptimizationLevel optLevel
- OptixCompileDebugLevel debugLevel
- const OptixModuleCompileBoundValueEntry ∗ boundValues
- unsigned int numBoundValues
- unsigned int numPayloadTypes
- OptixPayloadType ∗ payloadTypes

6.24.1 Detailed Description

Compilation options for module.

See Also

optixModuleCreateFromPTX()

6.24.2 Member Data Documentation

6.24.2.1 const OptixModuleCompileBoundValueEntry∗ OptixModuleCompileOptions::boundValues

Ingored if numBoundValues is set to 0.

6.24.2.2 OptixCompileDebugLevel OptixModuleCompileOptions::debugLevel

Generate debug information.

6.24.2.3 int OptixModuleCompileOptions::maxRegisterCount

Maximum number of registers allowed when compiling to SASS. Set to 0 for no explicit limit. May vary within a pipeline.

6.24.2.4 unsigned int OptixModuleCompileOptions::numBoundValues

set to 0 if unused

6.24.2.5 unsigned int OptixModuleCompileOptions::numPayloadTypes

The number of different payload types available for compilation. Must be zero if OptixPipelineCompileOptions::numPayloadValues is not zero.

6.24.2.6 OptixCompileOptimizationLevel OptixModuleCompileOptions::optLevel

Optimization level. May vary within a pipeline.

6.24.2.7 OptixPayloadType∗ OptixModuleCompileOptions::payloadTypes

Points to host array of payload type definitions, size must match numPayloadTypes.

6.25 OptixMotionOptions Struct Reference

Public Attributes

- unsigned short numKeys
- unsigned short flags
- float timeBegin
- float timeEnd

6.25.1 Detailed Description

Motion options.

See Also

OptixAccelBuildOptions::motionOptions, OptixMatrixMotionTransform::motionOptions, OptixSRTMotionTransform::motionOptions

6.25.2 Member Data Documentation

6.25.2.1 unsigned short OptixMotionOptions::flags

Combinations of OptixMotionFlags.

6.25.2.2 unsigned short OptixMotionOptions::numKeys

If numKeys $>$ 1, motion is enabled. timeBegin, timeEnd and flags are all ignored when motion is disabled.

6.25.2.3 float OptixMotionOptions::timeBegin

Point in time where motion starts.

6.25.2.4 float OptixMotionOptions::timeEnd

Point in time where motion ends.

6.26 OptixPayloadType Struct Reference

Public Attributes

- unsigned int numPayloadValues
- const unsigned int $*$ payloadSemantics

6.26.1 Detailed Description

Specifies a single payload type.

6.26.2 Member Data Documentation

6.26.2.1 unsigned int OptixPayloadType::numPayloadValues

The number of 32b words the payload of this type holds.

6.26.2.2 const unsigned int$*$ OptixPayloadType::payloadSemantics

Points to host array of payload word semantics, size must match numPayloadValues.

6.27 OptixPipelineCompileOptions Struct Reference

Public Attributes

- int usesMotionBlur
- unsigned int traversableGraphFlags
- int numPayloadValues
- int numAttributeValues
- unsigned int exceptionFlags
- const char ∗ pipelineLaunchParamsVariableName
- unsigned int usesPrimitiveTypeFlags

6.27.1 Detailed Description

Compilation options for all modules of a pipeline.

Similar to OptixModuleCompileOptions, but these options here need to be equal for all modules of a pipeline.

See Also

optixModuleCreateFromPTX(), optixPipelineCreate()

6.27.2 Member Data Documentation

6.27.2.1 unsigned int OptixPipelineCompileOptions::exceptionFlags

A bitmask of OptixExceptionFlags indicating which exceptions are enabled.

6.27.2.2 int OptixPipelineCompileOptions::numAttributeValues

How much storage, in 32b words, to make available for the attributes. The minimum number is 2. Values below that will automatically be changed to 2. [2..8].

6.27.2.3 int OptixPipelineCompileOptions::numPayloadValues

How much storage, in 32b words, to make available for the payload, [0..32] Must be zero if numPayloadTypes is not zero.

6.27.2.4 const char∗ OptixPipelineCompileOptions::pipelineLaunchParamsVariableName

The name of the pipeline parameter variable. If 0, no pipeline parameter will be available. This will be ignored if the launch param variable was optimized out or was not found in the modules linked to the pipeline.

6.27.2.5 unsigned int OptixPipelineCompileOptions::traversableGraphFlags

Traversable graph bitfield. See OptixTraversableGraphFlags.

6.27.2.6 int OptixPipelineCompileOptions::usesMotionBlur

Boolean value indicating whether motion blur could be used.

6.27.2.7 unsigned int OptixPipelineCompileOptions::usesPrimitiveTypeFlags

Bit field enabling primitive types. See OptixPrimitiveTypeFlags. Setting to zero corresponds to enabling OPTIX_PRIMITIVE_TYPE_FLAGS_CUSTOM and OPTIX_PRIMITIVE_TYPE_FLAGS_TRIANGLE.

6.28 OptixPipelineLinkOptions Struct Reference

Public Attributes

- unsigned int maxTraceDepth
- OptixCompileDebugLevel debugLevel

6.28.1 Detailed Description

Link options for a pipeline.

See Also

optixPipelineCreate()

6.28.2 Member Data Documentation

6.28.2.1 OptixCompileDebugLevel OptixPipelineLinkOptions::debugLevel

Generate debug information.

6.28.2.2 unsigned int OptixPipelineLinkOptions::maxTraceDepth

Maximum trace recursion depth. 0 means a ray generation program can be launched, but can't trace any rays. The maximum allowed value is 31.

6.29 OptixProgramGroupCallables Struct Reference

Public Attributes

- OptixModule moduleDC
- const char ∗ entryFunctionNameDC
- OptixModule moduleCC
- const char ∗ entryFunctionNameCC

6.29.1 Detailed Description

Program group representing callables.

Module and entry function name need to be valid for at least one of the two callables.

See Also

#OptixProgramGroupDesc::callables

6.29.2 Member Data Documentation

6.29.2.1 const char∗ OptixProgramGroupCallables::entryFunctionNameCC

Entry function name of the continuation callable (CC) program.

6.29.2.2 const char∗ OptixProgramGroupCallables::entryFunctionNameDC

Entry function name of the direct callable (DC) program.

6.29.2.3 OptixModule OptixProgramGroupCallables::moduleCC

Module holding the continuation callable (CC) program.

6.29.2.4 OptixModule OptixProgramGroupCallables::moduleDC

Module holding the direct callable (DC) program.

6.30 OptixProgramGroupDesc Struct Reference

Public Attributes

- OptixProgramGroupKind kind
- unsigned int flags
- union {
  OptixProgramGroupSingleModule raygen
  OptixProgramGroupSingleModule miss
  OptixProgramGroupSingleModule exception
  OptixProgramGroupCallables callables
  OptixProgramGroupHitgroup hitgroup
  };

6.30.1 Detailed Description

Descriptor for program groups.

6.30.2 Member Data Documentation

6.30.2.1 union { ... }

6.30.2.2 OptixProgramGroupCallables OptixProgramGroupDesc::callables

See Also

OPTIX_PROGRAM_GROUP_KIND_CALLABLES

6.30.2.3 OptixProgramGroupSingleModule OptixProgramGroupDesc::exception

See Also

OPTIX_PROGRAM_GROUP_KIND_EXCEPTION

6.30.2.4 unsigned int OptixProgramGroupDesc::flags

See OptixProgramGroupFlags.

6.30.2.5 OptixProgramGroupHitgroup OptixProgramGroupDesc::hitgroup

See Also

OPTIX_PROGRAM_GROUP_KIND_HITGROUP

6.30.2.6 OptixProgramGroupKind OptixProgramGroupDesc::kind

The kind of program group.

6.30.2.7 OptixProgramGroupSingleModule OptixProgramGroupDesc::miss

See Also

OPTIX_PROGRAM_GROUP_KIND_MISS

6.30.2.8 OptixProgramGroupSingleModule OptixProgramGroupDesc::raygen

See Also

OPTIX_PROGRAM_GROUP_KIND_RAYGEN

6.31 OptixProgramGroupHitgroup Struct Reference

Public Attributes

- OptixModule moduleCH
- const char ∗ entryFunctionNameCH
- OptixModule moduleAH
- const char ∗ entryFunctionNameAH
- OptixModule moduleIS
- const char ∗ entryFunctionNameIS

6.31.1 Detailed Description

Program group representing the hitgroup.

For each of the three program types, module and entry function name might both be `nullptr`.

See Also

OptixProgramGroupDesc::hitgroup

6.31.2 Member Data Documentation

6.31.2.1 const char∗ OptixProgramGroupHitgroup::entryFunctionNameAH

Entry function name of the any hit (AH) program.

6.31.2.2 const char∗ OptixProgramGroupHitgroup::entryFunctionNameCH

Entry function name of the closest hit (CH) program.

6.31.2.3 const char∗ OptixProgramGroupHitgroup::entryFunctionNameIS

Entry function name of the intersection (IS) program.

6.31.2.4 OptixModule OptixProgramGroupHitgroup::moduleAH

Module holding the any hit (AH) program.

6.31.2.5 OptixModule OptixProgramGroupHitgroup::moduleCH

Module holding the closest hit (CH) program.

6.31.2.6 OptixModule OptixProgramGroupHitgroup::moduleIS

Module holding the intersection (Is) program.

6.32 OptixProgramGroupOptions Struct Reference

Public Attributes

- OptixPayloadType ∗ payloadType

6.32.1 Detailed Description

Program group options.

See Also

optixProgramGroupCreate()

6.32.2 Member Data Documentation

6.32.2.1 OptixPayloadType∗ OptixProgramGroupOptions::payloadType

Specifies the payload type of this program group. All programs in the group must support the payload type (Program support for a type is specified by calling.

See Also

optixSetPayloadTypes or otherwise all types specified in OptixModuleCompileOptions are supported). If a program is not available for the requested payload type, optixProgramGroupCreate returns OPTIX_ERROR_PAYLOAD_TYPE_MISMATCH. If the payloadType is left zero, a unique type is deduced. The payload type can be uniquely deduced if there is exactly one payload type for which all programs in the group are available. If the payload type could not be deduced uniquely optixProgramGroupCreate returns OPTIX_ERROR_PAYLOAD_TYPE_RESOLUTION_FAILED.

6.33 OptixProgramGroupSingleModule Struct Reference

Public Attributes

- OptixModule module
- const char ∗ entryFunctionName

6.33.1 Detailed Description

Program group representing a single module.

Used for raygen, miss, and exception programs. In case of raygen and exception programs, module and entry function name need to be valid. For miss programs, module and entry function name might both be `nullptr`.

See Also

OptixProgramGroupDesc::raygen, OptixProgramGroupDesc::miss, OptixProgramGroupDesc::exception

6.33.2 Member Data Documentation

6.33.2.1 const char∗ OptixProgramGroupSingleModule::entryFunctionName

Entry function name of the single program.

6.33.2.2 OptixModule OptixProgramGroupSingleModule::module

Module holding single program.

6.34 OptixShaderBindingTable Struct Reference

Public Attributes

- CUdeviceptr raygenRecord
- CUdeviceptr exceptionRecord

- CUdeviceptr missRecordBase
- unsigned int missRecordStrideInBytes
- unsigned int missRecordCount

- CUdeviceptr hitgroupRecordBase
- unsigned int hitgroupRecordStrideInBytes
- unsigned int hitgroupRecordCount

- CUdeviceptr callablesRecordBase
- unsigned int callablesRecordStrideInBytes
- unsigned int callablesRecordCount

6.34.1 Detailed Description

Describes the shader binding table (SBT)

See Also

optixLaunch()

6.34.2 Member Data Documentation

6.34.2.1 CUdeviceptr OptixShaderBindingTable::callablesRecordBase

Arrays of SBT records for callable programs. If the base address is not null, the stride and count must not be zero. If the base address is null, then the count needs to zero. The base address and the stride must be a multiple of OPTIX_SBT_RECORD_ALIGNMENT.

6.34.2.2 unsigned int OptixShaderBindingTable::callablesRecordCount

Arrays of SBT records for callable programs. If the base address is not null, the stride and count must not be zero. If the base address is null, then the count needs to zero. The base address and the stride must be a multiple of OPTIX_SBT_RECORD_ALIGNMENT.

6.34.2.3 unsigned int OptixShaderBindingTable::callablesRecordStrideInBytes

Arrays of SBT records for callable programs. If the base address is not null, the stride and count must not be zero. If the base address is null, then the count needs to zero. The base address and the stride must be a multiple of OPTIX_SBT_RECORD_ALIGNMENT.

6.34.2.4 CUdeviceptr OptixShaderBindingTable::exceptionRecord

Device address of the SBT record of the exception program. The address must be a multiple of OPTIX_SBT_RECORD_ALIGNMENT.

6.34.2.5 CUdeviceptr OptixShaderBindingTable::hitgroupRecordBase

Arrays of SBT records for hit groups. The base address and the stride must be a multiple of OPTIX_SBT_RECORD_ALIGNMENT.

6.34.2.6 unsigned int OptixShaderBindingTable::hitgroupRecordCount

Arrays of SBT records for hit groups. The base address and the stride must be a multiple of OPTIX_SBT_RECORD_ALIGNMENT.

6.34.2.7 unsigned int OptixShaderBindingTable::hitgroupRecordStrideInBytes

Arrays of SBT records for hit groups. The base address and the stride must be a multiple of OPTIX_SBT_RECORD_ALIGNMENT.

6.34.2.8 CUdeviceptr OptixShaderBindingTable::missRecordBase

Arrays of SBT records for miss programs. The base address and the stride must be a multiple of OPTIX_SBT_RECORD_ALIGNMENT.

6.34.2.9 unsigned int OptixShaderBindingTable::missRecordCount

Arrays of SBT records for miss programs. The base address and the stride must be a multiple of OPTIX_SBT_RECORD_ALIGNMENT.

6.34.2.10 unsigned int OptixShaderBindingTable::missRecordStrideInBytes

Arrays of SBT records for miss programs. The base address and the stride must be a multiple of OPTIX_SBT_RECORD_ALIGNMENT.

6.34.2.11 CUdeviceptr OptixShaderBindingTable::raygenRecord

Device address of the SBT record of the ray gen program to start launch at. The address must be a multiple of OPTIX_SBT_RECORD_ALIGNMENT.

6.35 OptixSRTData Struct Reference

Public Attributes

Parameters describing the SRT transformation

- float sx
- float a
- float b
- float pvx
- float sy

- float c
- float pvy
- float sz
- float pvz
- float qx
- float qy
- float qz
- float qw
- float tx
- float ty
- float tz

6.35.1 Detailed Description

Represents an SRT transformation.

An SRT transformation can represent a smooth rotation with fewer motion keys than a matrix transformation. Each motion key is constructed from elements taken from a matrix S, a quaternion R, and a translation T.

The scaling matrix $S = \begin{bmatrix} sx & a & b & pvx \\ 0 & sy & c & pvy \\ 0 & 0 & sz & pvz \end{bmatrix}$ defines an affine transformation that can include scale, shear, and a translation. The translation allows to define the pivot point for the subsequent rotation.

The quaternion R = [qx, qy, qz, qw] describes a rotation with angular component qw = cos(theta/2) and other components [qx, qy, qz] = sin(theta/2) $*$ [ax, ay, az] where the axis [ax, ay, az] is normalized.

The translation matrix $T = \begin{bmatrix} 1 & 0 & 0 & tx \\ 0 & 1 & 0 & ty \\ 0 & 0 & 1 & tz \end{bmatrix}$ defines another translation that is applied after the rotation. Typically, this translation includes the inverse translation from the matrix S to reverse the translation for the pivot point for R.

To obtain the effective transformation at time t, the elements of the components of S, R, and T will be interpolated linearly. The components are then multiplied to obtain the combined transformation C = T $*$ R $*$ S. The transformation C is the effective object-to-world transformations at time t, and C^(-1) is the effective world-to-object transformation at time t.

See Also

OptixSRTMotionTransform::srtData, optixConvertPointerToTraversableHandle()

6.35.2 Member Data Documentation

6.35.2.1 float OptixSRTData::a

6.35.2.2 float OptixSRTData::b

6.35.2.3 float OptixSRTData::c

6.35.2.4 float OptixSRTData::pvx

6.35.2.5 float OptixSRTData::pvy

6.35.2.6 float OptixSRTData::pvz

6.35.2.7 float OptixSRTData::qw

6.35.2.8 float OptixSRTData::qx

6.35.2.9 float OptixSRTData::qy

6.35.2.10 float OptixSRTData::qz

6.35.2.11 float OptixSRTData::sx

6.35.2.12 float OptixSRTData::sy

6.35.2.13 float OptixSRTData::sz

6.35.2.14 float OptixSRTData::tx

6.35.2.15 float OptixSRTData::ty

6.35.2.16 float OptixSRTData::tz

6.36 OptixSRTMotionTransform Struct Reference

Public Attributes

- OptixTraversableHandle child
- OptixMotionOptions motionOptions
- unsigned int pad [3]
- OptixSRTData srtData [2]

6.36.1 Detailed Description

Represents an SRT motion transformation.

The device address of instances of this type must be a multiple of OPTIX_TRANSFORM_BYTE_ALIGNMENT.

This struct, as defined here, handles only N=2 motion keys due to the fixed array length of its srtData member. The following example shows how to create instances for an arbitrary number N of motion keys:

```
OptixSRTData srtData[N];
... // setup srtData

size_t transformSizeInBytes = sizeof( OptixSRTMotionTransform ) + ( N-2 ) * sizeof(
      OptixSRTData );
OptixSRTMotionTransform* srtMotionTransform = (
      OptixSRTMotionTransform*) malloc( transformSizeInBytes );
memset( srtMotionTransform, 0, transformSizeInBytes );

... // setup other members of srtMotionTransform
srtMotionTransform->motionOptions.numKeys   = N;
memcpy( srtMotionTransform->srtData, srtData, N * sizeof( OptixSRTData ) );

... // copy srtMotionTransform to device memory
free( srtMotionTransform )
```

See Also

optixConvertPointerToTraversableHandle()

6.36.2 Member Data Documentation

6.36.2.1 OptixTraversableHandle OptixSRTMotionTransform::child

The traversable transformed by this transformation.

6.36.2.2 OptixMotionOptions OptixSRTMotionTransform::motionOptions

The motion options for this transformation.

6.36.2.3 unsigned int OptixSRTMotionTransform::pad[3]

Padding to make the SRT data 16 byte aligned.

6.36.2.4 OptixSRTData OptixSRTMotionTransform::srtData[2]

The actual SRT data describing the transformation.

6.37 OptixStackSizes Struct Reference

Public Attributes

- unsigned int cssRG
- unsigned int cssMS
- unsigned int cssCH
- unsigned int cssAH
- unsigned int cssIS
- unsigned int cssCC
- unsigned int dssDC

6.37.1 Detailed Description

Describes the stack size requirements of a program group.

See Also

optixProgramGroupGetStackSize()

6.37.2 Member Data Documentation

6.37.2.1 unsigned int OptixStackSizes::cssAH

Continuation stack size of AH programs in bytes.

6.37.2.2 unsigned int OptixStackSizes::cssCC

Continuation stack size of CC programs in bytes.

6.37.2.3 unsigned int OptixStackSizes::cssCH

Continuation stack size of CH programs in bytes.

6.37.2.4 unsigned int OptixStackSizes::cssIS

Continuation stack size of IS programs in bytes.

6.37.2.5 unsigned int OptixStackSizes::cssMS

Continuation stack size of MS programs in bytes.

6.37.2.6 unsigned int OptixStackSizes::cssRG

Continuation stack size of RG programs in bytes.

6.37.2.7 unsigned int OptixStackSizes::dssDC

Direct stack size of DC programs in bytes.

6.38 OptixStaticTransform Struct Reference

Public Attributes

- OptixTraversableHandle child
- unsigned int pad [2]
- float transform [12]
- float invTransform [12]

6.38.1 Detailed Description

Static transform.

The device address of instances of this type must be a multiple of OPTIX_TRANSFORM_BYTE_ALIGNMENT.

See Also

optixConvertPointerToTraversableHandle()

6.38.2 Member Data Documentation

6.38.2.1 OptixTraversableHandle OptixStaticTransform::child

The traversable transformed by this transformation.

6.38.2.2 float OptixStaticTransform::invTransform[12]

Affine world-to-object transformation as 3x4 matrix in row-major layout Must be the inverse of the transform matrix.

6.38.2.3 unsigned int OptixStaticTransform::pad[2]

Padding to make the transformations 16 byte aligned.

6.38.2.4 float OptixStaticTransform::transform[12]

Affine object-to-world transformation as 3x4 matrix in row-major layout.

6.39 OptixUtilDenoiserImageTile Struct Reference

Public Attributes

- OptixImage2D input
- OptixImage2D output
- unsigned int inputOffsetX
- unsigned int inputOffsetY

6.39.1 Detailed Description

Tile definition.

see optixUtilDenoiserSplitImage

6.39.2 Member Data Documentation

6.39.2.1 OptixImage2D OptixUtilDenoiserImageTile::input

6.39.2.2 unsigned int OptixUtilDenoiserImageTile::inputOffsetX

6.39.2.3 unsigned int OptixUtilDenoiserImageTile::inputOffsetY

6.39.2.4 OptixImage2D OptixUtilDenoiserImageTile::output

6.40 optix_internal::TypePack<> Struct Template Reference

7 File Documentation

7.1 main.dox File Reference

7.2 optix.h File Reference

Macros

- #define OPTIX_VERSION 80000

7.2.1 Detailed Description

OptiX public API header.

Author

NVIDIA Corporation Includes the host api if compiling host code, includes the cuda api if compiling device code. For the math library routines include optix_math.h

7.2.2 Macro Definition Documentation

7.2.2.1 #define OPTIX_VERSION 80000

The OptiX version.

- major = OPTIX_VERSION/10000
- minor = (OPTIX_VERSION%10000)/100
- micro = OPTIX_VERSION%100

7.3 optix_7_device.h File Reference

Functions

- template<typename... Payload>
  static __forceinline__
  __device__ void optixTrace (OptixTraversableHandle handle, float3 rayOrigin, float3 rayDirection, float tmin, float tmax, float rayTime, OptixVisibilityMask visibilityMask, unsigned int rayFlags, unsigned int SBToffset, unsigned int SBTstride, unsigned int missSBTIndex, Payload &...payload)
- template<typename... Payload>
  static __forceinline__
  __device__ void optixTrace (OptixPayloadTypeID type, OptixTraversableHandle handle, float3 rayOrigin, float3 rayDirection, float tmin, float tmax, float rayTime, OptixVisibilityMask visibilityMask, unsigned int rayFlags, unsigned int SBToffset, unsigned int SBTstride, unsigned int missSBTIndex, Payload &...payload)
- static __forceinline__
  __device__ void optixSetPayload_0 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_1 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_2 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_3 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_4 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_5 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_6 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_7 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_8 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_9 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_10 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_11 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_12 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_13 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_14 (unsigned int p)

- static __forceinline__
  __device__ void optixSetPayload_15 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_16 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_17 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_18 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_19 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_20 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_21 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_22 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_23 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_24 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_25 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_26 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_27 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_28 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_29 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_30 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_31 (unsigned int p)
- static __forceinline__
  __device__ unsigned int optixGetPayload_0 ()
- static __forceinline__
  __device__ unsigned int optixGetPayload_1 ()
- static __forceinline__
  __device__ unsigned int optixGetPayload_2 ()
- static __forceinline__
  __device__ unsigned int optixGetPayload_3 ()
- static __forceinline__
  __device__ unsigned int optixGetPayload_4 ()

- static __forceinline__ __device__ unsigned int optixGetPayload_5 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_6 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_7 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_8 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_9 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_10 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_11 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_12 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_13 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_14 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_15 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_16 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_17 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_18 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_19 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_20 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_21 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_22 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_23 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_24 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_25 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_26 ()

- static __forceinline__
  __device__ unsigned int optixGetPayload_27 ()
- static __forceinline__
  __device__ unsigned int optixGetPayload_28 ()
- static __forceinline__
  __device__ unsigned int optixGetPayload_29 ()
- static __forceinline__
  __device__ unsigned int optixGetPayload_30 ()
- static __forceinline__
  __device__ unsigned int optixGetPayload_31 ()
- static __forceinline__
  __device__ void optixSetPayloadTypes (unsigned int typeMask)
- static __forceinline__
  __device__ unsigned int optixUndefinedValue ()
- static __forceinline__
  __device__ float3 optixGetWorldRayOrigin ()
- static __forceinline__
  __device__ float3 optixGetWorldRayDirection ()
- static __forceinline__
  __device__ float3 optixGetObjectRayOrigin ()
- static __forceinline__
  __device__ float3 optixGetObjectRayDirection ()
- static __forceinline__
  __device__ float optixGetRayTmin ()
- static __forceinline__
  __device__ float optixGetRayTmax ()
- static __forceinline__
  __device__ float optixGetRayTime ()
- static __forceinline__
  __device__ unsigned int optixGetRayFlags ()
- static __forceinline__
  __device__ unsigned int optixGetRayVisibilityMask ()
- static __forceinline__
  __device__
  OptixTraversableHandle optixGetInstanceTraversableFromIAS (OptixTraversableHandle ias, unsigned int instIdx)
- static __forceinline__
  __device__ void optixGetTriangleVertexData (OptixTraversableHandle gas, unsigned int primIdx, unsigned int sbtGASIndex, float time, float3 data[3])
- static __forceinline__
  __device__ void optixGetLinearCurveVertexData (OptixTraversableHandle gas, unsigned int primIdx, unsigned int sbtGASIndex, float time, float4 data[2])
- static __forceinline__
  __device__ void optixGetQuadraticBSplineVertexData (OptixTraversableHandle gas, unsigned int primIdx, unsigned int sbtGASIndex, float time, float4 data[3])

- static __forceinline__
  __device__ void optixGetCubicBSplineVertexData (OptixTraversableHandle gas, unsigned int primIdx, unsigned int sbtGASIndex, float time, float4 data[4])
- static __forceinline__
  __device__ void optixGetCatmullRomVertexData (OptixTraversableHandle gas, unsigned int primIdx, unsigned int sbtGASIndex, float time, float4 data[4])
- static __forceinline__
  __device__ void optixGetSphereData (OptixTraversableHandle gas, unsigned int primIdx, unsigned int sbtGASIndex, float time, float4 data[1])
- static __forceinline__
  __device__
  OptixTraversableHandle optixGetGASTraversableHandle ()
- static __forceinline__
  __device__ float optixGetGASMotionTimeBegin (OptixTraversableHandle gas)
- static __forceinline__
  __device__ float optixGetGASMotionTimeEnd (OptixTraversableHandle gas)
- static __forceinline__
  __device__ unsigned int optixGetGASMotionStepCount (OptixTraversableHandle gas)
- static __forceinline__
  __device__ void optixGetWorldToObjectTransformMatrix (float m[12])
- static __forceinline__
  __device__ void optixGetObjectToWorldTransformMatrix (float m[12])
- static __forceinline__
  __device__ float3 optixTransformPointFromWorldToObjectSpace (float3 point)
- static __forceinline__
  __device__ float3 optixTransformVectorFromWorldToObjectSpace (float3 vec)
- static __forceinline__
  __device__ float3 optixTransformNormalFromWorldToObjectSpace (float3 normal)
- static __forceinline__
  __device__ float3 optixTransformPointFromObjectToWorldSpace (float3 point)
- static __forceinline__
  __device__ float3 optixTransformVectorFromObjectToWorldSpace (float3 vec)
- static __forceinline__
  __device__ float3 optixTransformNormalFromObjectToWorldSpace (float3 normal)
- static __forceinline__
  __device__ unsigned int optixGetTransformListSize ()
- static __forceinline__
  __device__
  OptixTraversableHandle optixGetTransformListHandle (unsigned int index)
- static __forceinline__
  __device__ OptixTransformType optixGetTransformTypeFromHandle (OptixTraversableHandle handle)
- static __forceinline__
  __device__ const
  OptixStaticTransform ∗ optixGetStaticTransformFromHandle (OptixTraversableHandle handle)

- static __forceinline__ __device__ const OptixSRTMotionTransform ∗ optixGetSRTMotionTransformFromHandle (OptixTraversableHandle handle)
- static __forceinline__ __device__ const OptixMatrixMotionTransform ∗ optixGetMatrixMotionTransformFromHandle (OptixTraversableHandle handle)
- static __forceinline__ __device__ unsigned int optixGetInstanceIdFromHandle (OptixTraversableHandle handle)
- static __forceinline__ __device__ OptixTraversableHandle optixGetInstanceChildFromHandle (OptixTraversableHandle handle)
- static __forceinline__ __device__ const float4 ∗ optixGetInstanceTransformFromHandle (OptixTraversableHandle handle)
- static __forceinline__ __device__ const float4 ∗ optixGetInstanceInverseTransformFromHandle (OptixTraversableHandle handle)
- static __forceinline__ __device__ bool optixReportIntersection (float hitT, unsigned int hitKind)
- static __forceinline__ __device__ bool optixReportIntersection (float hitT, unsigned int hitKind, unsigned int a0)
- static __forceinline__ __device__ bool optixReportIntersection (float hitT, unsigned int hitKind, unsigned int a0, unsigned int a1)
- static __forceinline__ __device__ bool optixReportIntersection (float hitT, unsigned int hitKind, unsigned int a0, unsigned int a1, unsigned int a2)
- static __forceinline__ __device__ bool optixReportIntersection (float hitT, unsigned int hitKind, unsigned int a0, unsigned int a1, unsigned int a2, unsigned int a3)
- static __forceinline__ __device__ bool optixReportIntersection (float hitT, unsigned int hitKind, unsigned int a0, unsigned int a1, unsigned int a2, unsigned int a3, unsigned int a4)
- static __forceinline__ __device__ bool optixReportIntersection (float hitT, unsigned int hitKind, unsigned int a0, unsigned int a1, unsigned int a2, unsigned int a3, unsigned int a4, unsigned int a5)
- static __forceinline__ __device__ bool optixReportIntersection (float hitT, unsigned int hitKind, unsigned int a0, unsigned int a1, unsigned int a2, unsigned int a3, unsigned int a4, unsigned int a5, unsigned int a6)
- static __forceinline__ __device__ bool optixReportIntersection (float hitT, unsigned int hitKind, unsigned int a0, unsigned int a1, unsigned int a2, unsigned int a3, unsigned int a4, unsigned int a5, unsigned int a6, unsigned int a7)

- static __forceinline__
  __device__ unsigned int optixGetAttribute_0 ()
- static __forceinline__
  __device__ unsigned int optixGetAttribute_1 ()
- static __forceinline__
  __device__ unsigned int optixGetAttribute_2 ()
- static __forceinline__
  __device__ unsigned int optixGetAttribute_3 ()
- static __forceinline__
  __device__ unsigned int optixGetAttribute_4 ()
- static __forceinline__
  __device__ unsigned int optixGetAttribute_5 ()
- static __forceinline__
  __device__ unsigned int optixGetAttribute_6 ()
- static __forceinline__
  __device__ unsigned int optixGetAttribute_7 ()
- static __forceinline__
  __device__ void optixTerminateRay ()
- static __forceinline__
  __device__ void optixIgnoreIntersection ()
- static __forceinline__
  __device__ unsigned int optixGetPrimitiveIndex ()
- static __forceinline__
  __device__ unsigned int optixGetSbtGASIndex ()
- static __forceinline__
  __device__ unsigned int optixGetInstanceId ()
- static __forceinline__
  __device__ unsigned int optixGetInstanceIndex ()
- static __forceinline__
  __device__ unsigned int optixGetHitKind ()
- static __forceinline__
  __device__ OptixPrimitiveType optixGetPrimitiveType (unsigned int hitKind)
- static __forceinline__
  __device__ bool optixIsFrontFaceHit (unsigned int hitKind)
- static __forceinline__
  __device__ bool optixIsBackFaceHit (unsigned int hitKind)
- static __forceinline__
  __device__ OptixPrimitiveType optixGetPrimitiveType ()
- static __forceinline__
  __device__ bool optixIsFrontFaceHit ()
- static __forceinline__
  __device__ bool optixIsBackFaceHit ()
- static __forceinline__
  __device__ bool optixIsTriangleHit ()

- static __forceinline__
  __device__ bool optixIsTriangleFrontFaceHit ()
- static __forceinline__
  __device__ bool optixIsTriangleBackFaceHit ()
- static __forceinline__
  __device__ float2 optixGetTriangleBarycentrics ()
- static __forceinline__
  __device__ float optixGetCurveParameter ()
- static __forceinline__
  __device__ uint3 optixGetLaunchIndex ()
- static __forceinline__
  __device__ uint3 optixGetLaunchDimensions ()
- static __forceinline__
  __device__ CUdeviceptr optixGetSbtDataPointer ()
- static __forceinline__
  __device__ void optixThrowException (int exceptionCode)
- static __forceinline__
  __device__ void optixThrowException (int exceptionCode, unsigned int exceptionDetail0)
- static __forceinline__
  __device__ void optixThrowException (int exceptionCode, unsigned int exceptionDetail0, unsigned int exceptionDetail1)
- static __forceinline__
  __device__ void optixThrowException (int exceptionCode, unsigned int exceptionDetail0, unsigned int exceptionDetail1, unsigned int exceptionDetail2)
- static __forceinline__
  __device__ void optixThrowException (int exceptionCode, unsigned int exceptionDetail0, unsigned int exceptionDetail1, unsigned int exceptionDetail2, unsigned int exceptionDetail3)
- static __forceinline__
  __device__ void optixThrowException (int exceptionCode, unsigned int exceptionDetail0, unsigned int exceptionDetail1, unsigned int exceptionDetail2, unsigned int exceptionDetail3, unsigned int exceptionDetail4)
- static __forceinline__
  __device__ void optixThrowException (int exceptionCode, unsigned int exceptionDetail0, unsigned int exceptionDetail1, unsigned int exceptionDetail2, unsigned int exceptionDetail3, unsigned int exceptionDetail4, unsigned int exceptionDetail5)
- static __forceinline__
  __device__ void optixThrowException (int exceptionCode, unsigned int exceptionDetail0, unsigned int exceptionDetail1, unsigned int exceptionDetail2, unsigned int exceptionDetail3, unsigned int exceptionDetail4, unsigned int exceptionDetail5, unsigned int exceptionDetail6)
- static __forceinline__
  __device__ void optixThrowException (int exceptionCode, unsigned int exceptionDetail0, unsigned int exceptionDetail1, unsigned int exceptionDetail2, unsigned int exceptionDetail3, unsigned int exceptionDetail4, unsigned int exceptionDetail5, unsigned int exceptionDetail6, unsigned int exceptionDetail7)
- static __forceinline__
  __device__ int optixGetExceptionCode ()

- static __forceinline__
  __device__ unsigned int optixGetExceptionDetail_0 ()
- static __forceinline__
  __device__ unsigned int optixGetExceptionDetail_1 ()
- static __forceinline__
  __device__ unsigned int optixGetExceptionDetail_2 ()
- static __forceinline__
  __device__ unsigned int optixGetExceptionDetail_3 ()
- static __forceinline__
  __device__ unsigned int optixGetExceptionDetail_4 ()
- static __forceinline__
  __device__ unsigned int optixGetExceptionDetail_5 ()
- static __forceinline__
  __device__ unsigned int optixGetExceptionDetail_6 ()
- static __forceinline__
  __device__ unsigned int optixGetExceptionDetail_7 ()
- static __forceinline__
  __device__
  OptixTraversableHandle optixGetExceptionInvalidTraversable ()
- static __forceinline__
  __device__ int optixGetExceptionInvalidSbtOffset ()
- static __forceinline__
  __device__
  OptixInvalidRayExceptionDetails optixGetExceptionInvalidRay ()
- static __forceinline__
  __device__
  OptixParameterMismatchExceptionDetails optixGetExceptionParameterMismatch ()
- static __forceinline__
  __device__ char ∗ optixGetExceptionLineInfo ()
- template<typename ReturnT , typename... ArgTypes>
  static __forceinline__
  __device__ ReturnT optixDirectCall (unsigned int sbtIndex, ArgTypes...args)
- template<typename ReturnT , typename... ArgTypes>
  static __forceinline__
  __device__ ReturnT optixContinuationCall (unsigned int sbtIndex, ArgTypes...args)
- static __forceinline__
  __device__ uint4 optixTexFootprint2D (unsigned long long tex, unsigned int texInfo, float x, float y, unsigned int ∗singleMipLevel)
- static __forceinline__
  __device__ uint4 optixTexFootprint2DLod (unsigned long long tex, unsigned int texInfo, float x, float y, float level, bool coarse, unsigned int ∗singleMipLevel)
- static __forceinline__
  __device__ uint4 optixTexFootprint2DGrad (unsigned long long tex, unsigned int texInfo, float x, float y, float dPdx_x, float dPdx_y, float dPdy_x, float dPdy_y, bool coarse, unsigned int ∗singleMipLevel)

7.3.1 Detailed Description

OptiX public API header.

Author

NVIDIA Corporation OptiX public API Reference - Device API declarations

7.4 optix_7_device_impl.h File Reference

Classes

- struct optix_internal::TypePack<>

Namespaces

- optix_internal

Constant Groups

- optix_internal

Macros

- #define OPTIX_DEFINE_optixGetAttribute_BODY(which)
- #define OPTIX_DEFINE_optixGetExceptionDetail_BODY(which)

Functions

- template<typename... Payload>
  static __forceinline__
  __device__ void optixTrace (OptixTraversableHandle handle, float3 rayOrigin, float3 rayDirection, float tmin, float tmax, float rayTime, OptixVisibilityMask visibilityMask, unsigned int rayFlags, unsigned int SBToffset, unsigned int SBTstride, unsigned int missSBTIndex, Payload &...payload)
- template<typename... Payload>
  static __forceinline__
  __device__ void optixTrace (OptixPayloadTypeID type, OptixTraversableHandle handle, float3 rayOrigin, float3 rayDirection, float tmin, float tmax, float rayTime, OptixVisibilityMask visibilityMask, unsigned int rayFlags, unsigned int SBToffset, unsigned int SBTstride, unsigned int missSBTIndex, Payload &...payload)
- static __forceinline__
  __device__ void optixSetPayload_0 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_1 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_2 (unsigned int p)

- static __forceinline__
  __device__ void optixSetPayload_3 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_4 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_5 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_6 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_7 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_8 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_9 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_10 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_11 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_12 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_13 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_14 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_15 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_16 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_17 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_18 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_19 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_20 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_21 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_22 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_23 (unsigned int p)
- static __forceinline__
  __device__ void optixSetPayload_24 (unsigned int p)

- static __forceinline__ __device__ void optixSetPayload_25 (unsigned int p)
- static __forceinline__ __device__ void optixSetPayload_26 (unsigned int p)
- static __forceinline__ __device__ void optixSetPayload_27 (unsigned int p)
- static __forceinline__ __device__ void optixSetPayload_28 (unsigned int p)
- static __forceinline__ __device__ void optixSetPayload_29 (unsigned int p)
- static __forceinline__ __device__ void optixSetPayload_30 (unsigned int p)
- static __forceinline__ __device__ void optixSetPayload_31 (unsigned int p)
- static __forceinline__ __device__ unsigned int optixGetPayload_0 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_1 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_2 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_3 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_4 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_5 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_6 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_7 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_8 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_9 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_10 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_11 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_12 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_13 ()
- static __forceinline__ __device__ unsigned int optixGetPayload_14 ()

- static __forceinline__
  __device__ unsigned int optixGetPayload_15 ()
- static __forceinline__
  __device__ unsigned int optixGetPayload_16 ()
- static __forceinline__
  __device__ unsigned int optixGetPayload_17 ()
- static __forceinline__
  __device__ unsigned int optixGetPayload_18 ()
- static __forceinline__
  __device__ unsigned int optixGetPayload_19 ()
- static __forceinline__
  __device__ unsigned int optixGetPayload_20 ()
- static __forceinline__
  __device__ unsigned int optixGetPayload_21 ()
- static __forceinline__
  __device__ unsigned int optixGetPayload_22 ()
- static __forceinline__
  __device__ unsigned int optixGetPayload_23 ()
- static __forceinline__
  __device__ unsigned int optixGetPayload_24 ()
- static __forceinline__
  __device__ unsigned int optixGetPayload_25 ()
- static __forceinline__
  __device__ unsigned int optixGetPayload_26 ()
- static __forceinline__
  __device__ unsigned int optixGetPayload_27 ()
- static __forceinline__
  __device__ unsigned int optixGetPayload_28 ()
- static __forceinline__
  __device__ unsigned int optixGetPayload_29 ()
- static __forceinline__
  __device__ unsigned int optixGetPayload_30 ()
- static __forceinline__
  __device__ unsigned int optixGetPayload_31 ()
- static __forceinline__
  __device__ void optixSetPayloadTypes (unsigned int types)
- static __forceinline__
  __device__ unsigned int optixUndefinedValue ()
- static __forceinline__
  __device__ float3 optixGetWorldRayOrigin ()
- static __forceinline__
  __device__ float3 optixGetWorldRayDirection ()
- static __forceinline__
  __device__ float3 optixGetObjectRayOrigin ()

- static __forceinline__
  __device__ float3 optixGetObjectRayDirection ()
- static __forceinline__
  __device__ float optixGetRayTmin ()
- static __forceinline__
  __device__ float optixGetRayTmax ()
- static __forceinline__
  __device__ float optixGetRayTime ()
- static __forceinline__
  __device__ unsigned int optixGetRayFlags ()
- static __forceinline__
  __device__ unsigned int optixGetRayVisibilityMask ()
- static __forceinline__
  __device__
  OptixTraversableHandle optixGetInstanceTraversableFromIAS (OptixTraversableHandle ias, unsigned int instIdx)
- static __forceinline__
  __device__ void optixGetTriangleVertexData (OptixTraversableHandle gas, unsigned int primIdx, unsigned int sbtGASIndex, float time, float3 data[3])
- static __forceinline__
  __device__ void optixGetLinearCurveVertexData (OptixTraversableHandle gas, unsigned int primIdx, unsigned int sbtGASIndex, float time, float4 data[2])
- static __forceinline__
  __device__ void optixGetQuadraticBSplineVertexData (OptixTraversableHandle gas, unsigned int primIdx, unsigned int sbtGASIndex, float time, float4 data[3])
- static __forceinline__
  __device__ void optixGetCubicBSplineVertexData (OptixTraversableHandle gas, unsigned int primIdx, unsigned int sbtGASIndex, float time, float4 data[4])
- static __forceinline__
  __device__ void optixGetCatmullRomVertexData (OptixTraversableHandle gas, unsigned int primIdx, unsigned int sbtGASIndex, float time, float4 data[4])
- static __forceinline__
  __device__ void optixGetSphereData (OptixTraversableHandle gas, unsigned int primIdx, unsigned int sbtGASIndex, float time, float4 data[1])
- static __forceinline__
  __device__
  OptixTraversableHandle optixGetGASTraversableHandle ()
- static __forceinline__
  __device__ float optixGetGASMotionTimeBegin (OptixTraversableHandle handle)
- static __forceinline__
  __device__ float optixGetGASMotionTimeEnd (OptixTraversableHandle handle)
- static __forceinline__
  __device__ unsigned int optixGetGASMotionStepCount (OptixTraversableHandle handle)
- static __forceinline__
  __device__ void optixGetWorldToObjectTransformMatrix (float m[12])

- static __forceinline__
  __device__ void optixGetObjectToWorldTransformMatrix (float m[12])
- static __forceinline__
  __device__ float3 optixTransformPointFromWorldToObjectSpace (float3 point)
- static __forceinline__
  __device__ float3 optixTransformVectorFromWorldToObjectSpace (float3 vec)
- static __forceinline__
  __device__ float3 optixTransformNormalFromWorldToObjectSpace (float3 normal)
- static __forceinline__
  __device__ float3 optixTransformPointFromObjectToWorldSpace (float3 point)
- static __forceinline__
  __device__ float3 optixTransformVectorFromObjectToWorldSpace (float3 vec)
- static __forceinline__
  __device__ float3 optixTransformNormalFromObjectToWorldSpace (float3 normal)
- static __forceinline__
  __device__ unsigned int optixGetTransformListSize ()
- static __forceinline__
  __device__
  OptixTraversableHandle optixGetTransformListHandle (unsigned int index)
- static __forceinline__
  __device__ OptixTransformType optixGetTransformTypeFromHandle (OptixTraversableHandle handle)
- static __forceinline__
  __device__ const
  OptixStaticTransform ∗ optixGetStaticTransformFromHandle (OptixTraversableHandle handle)
- static __forceinline__
  __device__ const
  OptixSRTMotionTransform ∗ optixGetSRTMotionTransformFromHandle (OptixTraversableHandle handle)
- static __forceinline__
  __device__ const
  OptixMatrixMotionTransform ∗ optixGetMatrixMotionTransformFromHandle (OptixTraversableHandle handle)
- static __forceinline__
  __device__ unsigned int optixGetInstanceIdFromHandle (OptixTraversableHandle handle)
- static __forceinline__
  __device__
  OptixTraversableHandle optixGetInstanceChildFromHandle (OptixTraversableHandle handle)
- static __forceinline__
  __device__ const float4 ∗ optixGetInstanceTransformFromHandle (OptixTraversableHandle handle)
- static __forceinline__
  __device__ const float4 ∗ optixGetInstanceInverseTransformFromHandle (OptixTraversableHandle handle)
- static __forceinline__
  __device__ bool optixReportIntersection (float hitT, unsigned int hitKind)

- static __forceinline__
  __device__ bool optixReportIntersection (float hitT, unsigned int hitKind, unsigned int a0)
- static __forceinline__
  __device__ bool optixReportIntersection (float hitT, unsigned int hitKind, unsigned int a0, unsigned int a1)
- static __forceinline__
  __device__ bool optixReportIntersection (float hitT, unsigned int hitKind, unsigned int a0, unsigned int a1, unsigned int a2)
- static __forceinline__
  __device__ bool optixReportIntersection (float hitT, unsigned int hitKind, unsigned int a0, unsigned int a1, unsigned int a2, unsigned int a3)
- static __forceinline__
  __device__ bool optixReportIntersection (float hitT, unsigned int hitKind, unsigned int a0, unsigned int a1, unsigned int a2, unsigned int a3, unsigned int a4)
- static __forceinline__
  __device__ bool optixReportIntersection (float hitT, unsigned int hitKind, unsigned int a0, unsigned int a1, unsigned int a2, unsigned int a3, unsigned int a4, unsigned int a5)
- static __forceinline__
  __device__ bool optixReportIntersection (float hitT, unsigned int hitKind, unsigned int a0, unsigned int a1, unsigned int a2, unsigned int a3, unsigned int a4, unsigned int a5, unsigned int a6)
- static __forceinline__
  __device__ bool optixReportIntersection (float hitT, unsigned int hitKind, unsigned int a0, unsigned int a1, unsigned int a2, unsigned int a3, unsigned int a4, unsigned int a5, unsigned int a6, unsigned int a7)
- static __forceinline__
  __device__ unsigned int optixGetAttribute_0 ()
- static __forceinline__
  __device__ unsigned int optixGetAttribute_1 ()
- static __forceinline__
  __device__ unsigned int optixGetAttribute_2 ()
- static __forceinline__
  __device__ unsigned int optixGetAttribute_3 ()
- static __forceinline__
  __device__ unsigned int optixGetAttribute_4 ()
- static __forceinline__
  __device__ unsigned int optixGetAttribute_5 ()
- static __forceinline__
  __device__ unsigned int optixGetAttribute_6 ()
- static __forceinline__
  __device__ unsigned int optixGetAttribute_7 ()
- static __forceinline__
  __device__ void optixTerminateRay ()
- static __forceinline__
  __device__ void optixIgnoreIntersection ()

- static __forceinline__
  __device__ unsigned int optixGetPrimitiveIndex ()
- static __forceinline__
  __device__ unsigned int optixGetSbtGASIndex ()
- static __forceinline__
  __device__ unsigned int optixGetInstanceId ()
- static __forceinline__
  __device__ unsigned int optixGetInstanceIndex ()
- static __forceinline__
  __device__ unsigned int optixGetHitKind ()
- static __forceinline__
  __device__ OptixPrimitiveType optixGetPrimitiveType (unsigned int hitKind)
- static __forceinline__
  __device__ bool optixIsBackFaceHit (unsigned int hitKind)
- static __forceinline__
  __device__ bool optixIsFrontFaceHit (unsigned int hitKind)
- static __forceinline__
  __device__ OptixPrimitiveType optixGetPrimitiveType ()
- static __forceinline__
  __device__ bool optixIsBackFaceHit ()
- static __forceinline__
  __device__ bool optixIsFrontFaceHit ()
- static __forceinline__
  __device__ bool optixIsTriangleHit ()
- static __forceinline__
  __device__ bool optixIsTriangleFrontFaceHit ()
- static __forceinline__
  __device__ bool optixIsTriangleBackFaceHit ()
- static __forceinline__
  __device__ float optixGetCurveParameter ()
- static __forceinline__
  __device__ float2 optixGetTriangleBarycentrics ()
- static __forceinline__
  __device__ uint3 optixGetLaunchIndex ()
- static __forceinline__
  __device__ uint3 optixGetLaunchDimensions ()
- static __forceinline__
  __device__ CUdeviceptr optixGetSbtDataPointer ()
- static __forceinline__
  __device__ void optixThrowException (int exceptionCode)
- static __forceinline__
  __device__ void optixThrowException (int exceptionCode, unsigned int exceptionDetail0)
- static __forceinline__
  __device__ void optixThrowException (int exceptionCode, unsigned int exceptionDetail0, unsigned int exceptionDetail1)

- static __forceinline__
  __device__ void optixThrowException (int exceptionCode, unsigned int exceptionDetail0, unsigned int exceptionDetail1, unsigned int exceptionDetail2)
- static __forceinline__
  __device__ void optixThrowException (int exceptionCode, unsigned int exceptionDetail0, unsigned int exceptionDetail1, unsigned int exceptionDetail2, unsigned int exceptionDetail3)
- static __forceinline__
  __device__ void optixThrowException (int exceptionCode, unsigned int exceptionDetail0, unsigned int exceptionDetail1, unsigned int exceptionDetail2, unsigned int exceptionDetail3, unsigned int exceptionDetail4)
- static __forceinline__
  __device__ void optixThrowException (int exceptionCode, unsigned int exceptionDetail0, unsigned int exceptionDetail1, unsigned int exceptionDetail2, unsigned int exceptionDetail3, unsigned int exceptionDetail4, unsigned int exceptionDetail5)
- static __forceinline__
  __device__ void optixThrowException (int exceptionCode, unsigned int exceptionDetail0, unsigned int exceptionDetail1, unsigned int exceptionDetail2, unsigned int exceptionDetail3, unsigned int exceptionDetail4, unsigned int exceptionDetail5, unsigned int exceptionDetail6)
- static __forceinline__
  __device__ void optixThrowException (int exceptionCode, unsigned int exceptionDetail0, unsigned int exceptionDetail1, unsigned int exceptionDetail2, unsigned int exceptionDetail3, unsigned int exceptionDetail4, unsigned int exceptionDetail5, unsigned int exceptionDetail6, unsigned int exceptionDetail7)
- static __forceinline__
  __device__ int optixGetExceptionCode ()
- static __forceinline__
  __device__ unsigned int optixGetExceptionDetail_0 ()
- static __forceinline__
  __device__ unsigned int optixGetExceptionDetail_1 ()
- static __forceinline__
  __device__ unsigned int optixGetExceptionDetail_2 ()
- static __forceinline__
  __device__ unsigned int optixGetExceptionDetail_3 ()
- static __forceinline__
  __device__ unsigned int optixGetExceptionDetail_4 ()
- static __forceinline__
  __device__ unsigned int optixGetExceptionDetail_5 ()
- static __forceinline__
  __device__ unsigned int optixGetExceptionDetail_6 ()
- static __forceinline__
  __device__ unsigned int optixGetExceptionDetail_7 ()
- static __forceinline__
  __device__
  OptixTraversableHandle optixGetExceptionInvalidTraversable ()
- static __forceinline__
  __device__ int optixGetExceptionInvalidSbtOffset ()

- static __forceinline__
  __device__
  OptixInvalidRayExceptionDetails optixGetExceptionInvalidRay ()
- static __forceinline__
  __device__
  OptixParameterMismatchExceptionDetails optixGetExceptionParameterMismatch ()
- static __forceinline__
  __device__ char ∗ optixGetExceptionLineInfo ()
- template<typename ReturnT , typename... ArgTypes>
  static __forceinline__
  __device__ ReturnT optixDirectCall (unsigned int sbtIndex, ArgTypes...args)
- template<typename ReturnT , typename... ArgTypes>
  static __forceinline__
  __device__ ReturnT optixContinuationCall (unsigned int sbtIndex, ArgTypes...args)
- static __forceinline__
  __device__ uint4 optixTexFootprint2D (unsigned long long tex, unsigned int texInfo, float x, float y, unsigned int ∗singleMipLevel)
- static __forceinline__
  __device__ uint4 optixTexFootprint2DGrad (unsigned long long tex, unsigned int texInfo, float x, float y, float dPdx_x, float dPdx_y, float dPdy_x, float dPdy_y, bool coarse, unsigned int ∗singleMipLevel)
- static __forceinline__
  __device__ uint4 optixTexFootprint2DLod (unsigned long long tex, unsigned int texInfo, float x, float y, float level, bool coarse, unsigned int ∗singleMipLevel)

7.4.1 Detailed Description

OptiX public API.

Author

NVIDIA Corporation OptiX public API Reference - Device side implementation

7.4.2 Macro Definition Documentation

7.4.2.1 #define OPTIX_DEFINE_optixGetAttribute_BODY(*which*)

Value:

```
unsigned int ret;
             \
    asm( "call (%0), _optix_get_attribute_" #which ", ();" : "=r"( ret ) : );
                 \
    return ret;
```

7.4.2.2 #define OPTIX_DEFINE_optixGetExceptionDetail_BODY(*which*)

Value:

```
unsigned int ret;
            \
    asm( "call (%0), _optix_get_exception_detail_" #which ", ();" : "=r"( ret ) : );
                \
    return ret;
```

7.4.3 Function Documentation

7.4.3.1 template<typename ReturnT , typename... ArgTypes> static __forceinline__ __device__ ReturnT optixContinuationCall (unsigned int *sbtIndex,* ArgTypes... *args*) `[static]`

7.4.3.2 template<typename ReturnT , typename... ArgTypes> static __forceinline__ __device__ ReturnT optixDirectCall (unsigned int *sbtIndex,* ArgTypes... *args*) `[static]`

7.4.3.3 static __forceinline__ __device__ unsigned int optixGetAttribute_0 ( ) `[static]`

7.4.3.4 static __forceinline__ __device__ unsigned int optixGetAttribute_1 ( ) `[static]`

7.4.3.5 static __forceinline__ __device__ unsigned int optixGetAttribute_2 ( ) `[static]`

7.4.3.6 static __forceinline__ __device__ unsigned int optixGetAttribute_3 ( ) `[static]`

7.4.3.7 static __forceinline__ __device__ unsigned int optixGetAttribute_4 ( ) `[static]`

7.4.3.8 static __forceinline__ __device__ unsigned int optixGetAttribute_5 ( ) `[static]`

7.4.3.9 static __forceinline__ __device__ unsigned int optixGetAttribute_6 ( ) `[static]`

7.4.3.10 static __forceinline__ __device__ unsigned int optixGetAttribute_7 ( ) `[static]`

7.4.3.11 static __forceinline__ __device__ void optixGetCatmullRomVertexData (OptixTraversableHandle *gas,* unsigned int *primIdx,* unsigned int *sbtGASIndex,* float *time,*

float4 ***data[4]***) `[static]`

7.4.3.12 static __forceinline__ __device__ void optixGetCubicBSplineVertexData (

OptixTraversableHandle ***gas,***
unsigned int ***primIdx,***
unsigned int ***sbtGASIndex,***
float ***time,***

float4 *data[4]*) `[static]`

7.4.3.13 static __forceinline__ __device__ float optixGetCurveParameter ( ) `[static]`

7.4.3.14 static __forceinline__ __device__ int optixGetExceptionCode ( ) `[static]`

7.4.3.15 static __forceinline__ __device__ unsigned int optixGetExceptionDetail_0 ( ) `[static]`

7.4.3.16 static __forceinline__ __device__ unsigned int optixGetExceptionDetail_1 ( ) `[static]`

7.4.3.17 static __forceinline__ __device__ unsigned int optixGetExceptionDetail_2 ( ) `[static]`

7.4.3.18 static __forceinline__ __device__ unsigned int optixGetExceptionDetail_3 ( ) `[static]`

7.4.3.19 static __forceinline__ __device__ unsigned int optixGetExceptionDetail_4 ( ) `[static]`

7.4.3.20 static __forceinline__ __device__ unsigned int optixGetExceptionDetail_5 ( ) `[static]`

7.4.3.21 static __forceinline__ __device__ unsigned int optixGetExceptionDetail_6 ( ) `[static]`

7.4.3.22 static __forceinline__ __device__ unsigned int optixGetExceptionDetail_7 ( ) `[static]`

7.4.3.23 static __forceinline__ __device__ OptixInvalidRayExceptionDetails optixGetExceptionInvalidRay ( ) `[static]`

7.4.3.24 static __forceinline__ __device__ int optixGetExceptionInvalidSbtOffset ( ) `[static]`

7.4.3.25 static __forceinline__ __device__ OptixTraversableHandle optixGetExceptionInvalidTraversable ( ) `[static]`

7.4.3.26 static __forceinline__ __device__ char∗ optixGetExceptionLineInfo ( ) `[static]`

7.4.3.27 static __forceinline__ __device__ OptixParameterMismatchExceptionDetails optixGetExceptionParameterMismatch ( ) `[static]`

7.4.3.28 static __forceinline__ __device__ unsigned int optixGetGASMotionStepCount (OptixTraversableHandle *handle*) `[static]`

7.4.3.29 static __forceinline__ __device__ float optixGetGASMotionTimeBegin (

OptixTraversableHandle *handle*) `[static]`

7.4.3.30 static __forceinline__ __device__ float optixGetGASMotionTimeEnd (OptixTraversableHandle *handle*) `[static]`

7.4.3.31 static __forceinline__ __device__ OptixTraversableHandle optixGetGASTraversableHandle ( ) `[static]`

7.4.3.32 static __forceinline__ __device__ unsigned int optixGetHitKind ( ) `[static]`

7.4.3.33 static __forceinline__ __device__ OptixTraversableHandle optixGetInstanceChildFromHandle (OptixTraversableHandle *handle*) `[static]`

7.4.3.34 static __forceinline__ __device__ unsigned int optixGetInstanceId ( ) `[static]`

7.4.3.35 static __forceinline__ __device__ unsigned int optixGetInstanceIdFromHandle (OptixTraversableHandle *handle*) `[static]`

7.4.3.36 static __forceinline__ __device__ unsigned int optixGetInstanceIndex ( ) `[static]`

7.4.3.37 static __forceinline__ __device__ const float4∗ optixGetInstanceInverseTransform-FromHandle (OptixTraversableHandle *handle*) `[static]`

7.4.3.38 static __forceinline__ __device__ const float4∗ optixGetInstanceTransformFromHandle (OptixTraversableHandle *handle*) `[static]`

7.4.3.39 static __forceinline__ __device__ OptixTraversableHandle optixGetInstanceTraversableFromIAS (OptixTraversableHandle *ias,* unsigned int *instIdx*) `[static]`

7.4.3.40 static __forceinline__ __device__ uint3 optixGetLaunchDimensions ( ) `[static]`

7.4.3.41 static __forceinline__ __device__ uint3 optixGetLaunchIndex ( ) `[static]`

7.4.3.42 static __forceinline__ __device__ void optixGetLinearCurveVertexData (OptixTraversableHandle *gas,* unsigned int *primIdx,* unsigned int *sbtGASIndex,* float *time,* float4 *data[2]*) `[static]`

7.4.3.43 static __forceinline__ __device__ const OptixMatrixMotionTransform∗ optixGetMatrixMotionTransformFromHandle (

OptixTraversableHandle *handle*) [static]

7.4.3.44 static __forceinline__ __device__ float3 optixGetObjectRayDirection ( ) [static]

7.4.3.45 static __forceinline__ __device__ float3 optixGetObjectRayOrigin ( ) [static]

7.4.3.46 static __forceinline__ __device__ void optixGetObjectToWorldTransformMatrix (

float *m[12]*) `[static]`

7.4.3.47 static __forceinline__ __device__ unsigned int optixGetPayload_0 ( ) `[static]`

7.4.3.48 static __forceinline__ __device__ unsigned int optixGetPayload_1 ( ) `[static]`

7.4.3.49 static __forceinline__ __device__ unsigned int optixGetPayload_10 ( ) `[static]`

7.4.3.50 static __forceinline__ __device__ unsigned int optixGetPayload_11 ( ) `[static]`

7.4.3.51 static __forceinline__ __device__ unsigned int optixGetPayload_12 ( ) `[static]`

7.4.3.52 static __forceinline__ __device__ unsigned int optixGetPayload_13 ( ) `[static]`

7.4.3.53 static __forceinline__ __device__ unsigned int optixGetPayload_14 ( ) `[static]`

7.4.3.54 static __forceinline__ __device__ unsigned int optixGetPayload_15 ( ) `[static]`

7.4.3.55 static __forceinline__ __device__ unsigned int optixGetPayload_16 ( ) `[static]`

7.4.3.56 static __forceinline__ __device__ unsigned int optixGetPayload_17 ( ) `[static]`

7.4.3.57 static __forceinline__ __device__ unsigned int optixGetPayload_18 ( ) `[static]`

7.4.3.58 static __forceinline__ __device__ unsigned int optixGetPayload_19 ( ) `[static]`

7.4.3.59 static __forceinline__ __device__ unsigned int optixGetPayload_2 ( ) `[static]`

7.4.3.60 static __forceinline__ __device__ unsigned int optixGetPayload_20 ( ) `[static]`

7.4.3.61 static __forceinline__ __device__ unsigned int optixGetPayload_21 ( ) `[static]`

7.4.3.62 static __forceinline__ __device__ unsigned int optixGetPayload_22 ( ) `[static]`

7.4.3.63 static __forceinline__ __device__ unsigned int optixGetPayload_23 ( ) `[static]`

7.4.3.64 static __forceinline__ __device__ unsigned int optixGetPayload_24 ( ) `[static]`

7.4.3.65 static __forceinline__ __device__ unsigned int optixGetPayload_25 ( ) `[static]`

7.4.3.66 static __forceinline__ __device__ unsigned int optixGetPayload_26 ( ) `[static]`

7.4.3.67 static __forceinline__ __device__ unsigned int optixGetPayload_27 ( ) `[static]`

7.4.3.68 static __forceinline__ __device__ unsigned int optixGetPayload_28 ( ) `[static]`

7.4.3.69 static __forceinline__ __device__ unsigned int optixGetPayload_29 ( ) `[static]`

7.4.3.70 static __forceinline__ __device__ unsigned int optixGetPayload_3 ( ) `[static]`

7.4.3.71 static __forceinline__ __device__ unsigned int optixGetPayload_30 ( ) `[static]`

7.4.3.72 static __forceinline__ __device__ unsigned int optixGetPayload_31 ( ) `[static]`

7.4.3.73 static __forceinline__ __device__ unsigned int optixGetPayload_4 ( ) `[static]`

unsigned int *hitKind*) `[static]`

7.4.3.81 static __forceinline__ __device__ OptixPrimitiveType optixGetPrimitiveType ( ) `[static]`

7.4.3.82 static __forceinline__ __device__ void optixGetQuadraticBSplineVertexData (OptixTraversableHandle *gas*, unsigned int *primIdx*, unsigned int *sbtGASIndex*, float *time*, float4 *data[3]*) `[static]`

7.4.3.83 static __forceinline__ __device__ unsigned int optixGetRayFlags ( ) `[static]`

7.4.3.84 static __forceinline__ __device__ float optixGetRayTime ( ) `[static]`

7.4.3.85 static __forceinline__ __device__ float optixGetRayTmax ( ) `[static]`

7.4.3.86 static __forceinline__ __device__ float optixGetRayTmin ( ) `[static]`

7.4.3.87 static __forceinline__ __device__ unsigned int optixGetRayVisibilityMask ( ) `[static]`

7.4.3.88 static __forceinline__ __device__ CUdeviceptr optixGetSbtDataPointer ( ) `[static]`

7.4.3.89 static __forceinline__ __device__ unsigned int optixGetSbtGASIndex ( ) `[static]`

7.4.3.90 static __forceinline__ __device__ void optixGetSphereData (OptixTraversableHandle *gas*, unsigned int *primIdx*, unsigned int *sbtGASIndex*, float *time*, float4 *data[1]*) `[static]`

7.4.3.91 static __forceinline__ __device__ const OptixSRTMotionTransform* optixGetSRTMotionTransformFromHandle (OptixTraversableHandle *handle*) `[static]`

7.4.3.92 static __forceinline__ __device__ const OptixStaticTransform* optixGetStaticTransformFromHandle (OptixTraversableHandle *handle*) `[static]`

7.4.3.93 static __forceinline__ __device__ OptixTraversableHandle optixGetTransformListHandle (

unsigned int *index*) `[static]`

7.4.3.94 static __forceinline__ __device__ unsigned int optixGetTransformListSize ( ) `[static]`

7.4.3.95 static __forceinline__ __device__ OptixTransformType optixGetTransformTypeFromHandle (

OptixTraversableHandle *handle*) `[static]`

7.4.3.96 static __forceinline__ __device__ float2 optixGetTriangleBarycentrics ( ) `[static]`

7.4.3.97 static __forceinline__ __device__ void optixGetTriangleVertexData (

OptixTraversableHandle *gas,*
unsigned int *primIdx,*
unsigned int *sbtGASIndex,*
float *time,*
float3 *data[3]*) `[static]`

7.4.3.98 static __forceinline__ __device__ float3 optixGetWorldRayDirection ( ) `[static]`

7.4.3.99 static __forceinline__ __device__ float3 optixGetWorldRayOrigin ( ) `[static]`

7.4.3.100 static __forceinline__ __device__ void optixGetWorldToObjectTransformMatrix (

float *m[12]*) `[static]`

7.4.3.101 static __forceinline__ __device__ void optixIgnoreIntersection ( ) `[static]`

7.4.3.102 static __forceinline__ __device__ bool optixIsBackFaceHit (

unsigned int *hitKind*) `[static]`

7.4.3.103 static __forceinline__ __device__ bool optixIsBackFaceHit ( ) `[static]`

7.4.3.104 static __forceinline__ __device__ bool optixIsFrontFaceHit (

unsigned int *hitKind*) `[static]`

7.4.3.105 static __forceinline__ __device__ bool optixIsFrontFaceHit ( ) `[static]`

7.4.3.106 static __forceinline__ __device__ bool optixIsTriangleBackFaceHit ( ) `[static]`

7.4.3.107 static __forceinline__ __device__ bool optixIsTriangleFrontFaceHit ( ) `[static]`

7.4.3.108 static __forceinline__ __device__ bool optixIsTriangleHit ( ) `[static]`

7.4.3.109 static __forceinline__ __device__ bool optixReportIntersection (

float *hitT,*
unsigned int *hitKind*) `[static]`

7.4.3.110 static __forceinline__ __device__ bool optixReportIntersection (

float *hitT,*
unsigned int *hitKind,*
unsigned int *a0*) `[static]`

7.4.3.111 static __forceinline__ __device__ bool optixReportIntersection (

float *hitT,*
unsigned int *hitKind,*
unsigned int *a0,*
unsigned int *a1*) `[static]`

7.4.3.112 static __forceinline__ __device__ bool optixReportIntersection (

float *hitT,*
unsigned int *hitKind,*
unsigned int *a0,*
unsigned int *a1,*
unsigned int *a2*) `[static]`

7.4.3.113 static __forceinline__ __device__ bool optixReportIntersection (

float *hitT,*
unsigned int *hitKind,*
unsigned int *a0,*
unsigned int *a1,*
unsigned int *a2,*
unsigned int *a3*) `[static]`

7.4.3.114 static __forceinline__ __device__ bool optixReportIntersection (

float *hitT,*
unsigned int *hitKind,*
unsigned int *a0,*
unsigned int *a1,*
unsigned int *a2,*
unsigned int *a3,*
unsigned int *a4*) `[static]`

7.4.3.115 static __forceinline__ __device__ bool optixReportIntersection (

float *hitT,*
unsigned int *hitKind,*
unsigned int *a0,*
unsigned int *a1,*
unsigned int *a2,*
unsigned int *a3,*
unsigned int *a4,*

unsigned int *a5*) `[static]`

7.4.3.116 static __forceinline__ __device__ bool optixReportIntersection (float *hitT*, unsigned int *hitKind*, unsigned int *a0*, unsigned int *a1*, unsigned int *a2*, unsigned int *a3*, unsigned int *a4*, unsigned int *a5*, unsigned int *a6*) `[static]`

7.4.3.117 static __forceinline__ __device__ bool optixReportIntersection (float *hitT*, unsigned int *hitKind*, unsigned int *a0*, unsigned int *a1*, unsigned int *a2*, unsigned int *a3*, unsigned int *a4*, unsigned int *a5*, unsigned int *a6*, unsigned int *a7*) `[static]`

7.4.3.118 static __forceinline__ __device__ void optixSetPayload_0 (unsigned int *p*) `[static]`

7.4.3.119 static __forceinline__ __device__ void optixSetPayload_1 (unsigned int *p*) `[static]`

7.4.3.120 static __forceinline__ __device__ void optixSetPayload_10 (unsigned int *p*) `[static]`

7.4.3.121 static __forceinline__ __device__ void optixSetPayload_11 (unsigned int *p*) `[static]`

7.4.3.122 static __forceinline__ __device__ void optixSetPayload_12 (unsigned int *p*) `[static]`

7.4.3.123 static __forceinline__ __device__ void optixSetPayload_13 (unsigned int *p*) `[static]`

7.4.3.124 static __forceinline__ __device__ void optixSetPayload_14 (

unsigned int *p*) `[static]`

7.4.3.125 static __forceinline__ __device__ void optixSetPayload_15 (unsigned int *p*) `[static]`

7.4.3.126 static __forceinline__ __device__ void optixSetPayload_16 (unsigned int *p*) `[static]`

7.4.3.127 static __forceinline__ __device__ void optixSetPayload_17 (unsigned int *p*) `[static]`

7.4.3.128 static __forceinline__ __device__ void optixSetPayload_18 (unsigned int *p*) `[static]`

7.4.3.129 static __forceinline__ __device__ void optixSetPayload_19 (unsigned int *p*) `[static]`

7.4.3.130 static __forceinline__ __device__ void optixSetPayload_2 (unsigned int *p*) `[static]`

7.4.3.131 static __forceinline__ __device__ void optixSetPayload_20 (unsigned int *p*) `[static]`

7.4.3.132 static __forceinline__ __device__ void optixSetPayload_21 (unsigned int *p*) `[static]`

7.4.3.133 static __forceinline__ __device__ void optixSetPayload_22 (unsigned int *p*) `[static]`

7.4.3.134 static __forceinline__ __device__ void optixSetPayload_23 (unsigned int *p*) `[static]`

7.4.3.135 static __forceinline__ __device__ void optixSetPayload_24 (unsigned int *p*) `[static]`

7.4.3.136 static __forceinline__ __device__ void optixSetPayload_25 (unsigned int *p*) `[static]`

7.4.3.137 static __forceinline__ __device__ void optixSetPayload_26 (unsigned int *p*) `[static]`

7.4.3.138 static __forceinline__ __device__ void optixSetPayload_27 (unsigned int *p*) `[static]`

7.4.3.139 static __forceinline__ __device__ void optixSetPayload_28 (

unsigned int *p*) `[static]`

7.4.3.140 static __forceinline__ __device__ void optixSetPayload_29 (unsigned int *p*) `[static]`

7.4.3.141 static __forceinline__ __device__ void optixSetPayload_3 (unsigned int *p*) `[static]`

7.4.3.142 static __forceinline__ __device__ void optixSetPayload_30 (unsigned int *p*) `[static]`

7.4.3.143 static __forceinline__ __device__ void optixSetPayload_31 (unsigned int *p*) `[static]`

7.4.3.144 static __forceinline__ __device__ void optixSetPayload_4 (unsigned int *p*) `[static]`

7.4.3.145 static __forceinline__ __device__ void optixSetPayload_5 (unsigned int *p*) `[static]`

7.4.3.146 static __forceinline__ __device__ void optixSetPayload_6 (unsigned int *p*) `[static]`

7.4.3.147 static __forceinline__ __device__ void optixSetPayload_7 (unsigned int *p*) `[static]`

7.4.3.148 static __forceinline__ __device__ void optixSetPayload_8 (unsigned int *p*) `[static]`

7.4.3.149 static __forceinline__ __device__ void optixSetPayload_9 (unsigned int *p*) `[static]`

7.4.3.150 static __forceinline__ __device__ void optixSetPayloadTypes (unsigned int *types*) `[static]`

7.4.3.151 static __forceinline__ __device__ void optixTerminateRay ( ) `[static]`

7.4.3.152 static __forceinline__ __device__ uint4 optixTexFootprint2D (

unsigned long long *tex,*
unsigned int *texInfo,*
float *x,*
float *y,*
unsigned int ∗ *singleMipLevel*) `[static]`

7.4.3.153 static __forceinline__ __device__ uint4 optixTexFootprint2DGrad (

unsigned long long *tex,*
unsigned int *texInfo,*

float *x,*
float *y,*
float *dPdx_x,*
float *dPdx_y,*
float *dPdy_x,*
float *dPdy_y,*
bool *coarse,*
unsigned int ∗ *singleMipLevel*) `[static]`

7.4.3.154 static __forceinline__ __device__ uint4 optixTexFootprint2DLod (

unsigned long long *tex,*
unsigned int *texInfo,*
float *x,*
float *y,*
float *level,*
bool *coarse,*
unsigned int ∗ *singleMipLevel*) `[static]`

7.4.3.155 static __forceinline__ __device__ void optixThrowException (

int *exceptionCode*) `[static]`

7.4.3.156 static __forceinline__ __device__ void optixThrowException (

int *exceptionCode,*
unsigned int *exceptionDetail0*) `[static]`

7.4.3.157 static __forceinline__ __device__ void optixThrowException (

int *exceptionCode,*
unsigned int *exceptionDetail0,*
unsigned int *exceptionDetail1*) `[static]`

7.4.3.158 static __forceinline__ __device__ void optixThrowException (

int *exceptionCode,*
unsigned int *exceptionDetail0,*
unsigned int *exceptionDetail1,*
unsigned int *exceptionDetail2*) `[static]`

7.4.3.159 static __forceinline__ __device__ void optixThrowException (

int *exceptionCode,*
unsigned int *exceptionDetail0,*
unsigned int *exceptionDetail1,*
unsigned int *exceptionDetail2,*
unsigned int *exceptionDetail3*) `[static]`

7.4.3.160 static __forceinline__ __device__ void optixThrowException (

int *exceptionCode,*
unsigned int *exceptionDetail0,*
unsigned int *exceptionDetail1,*
unsigned int *exceptionDetail2,*
unsigned int *exceptionDetail3,*
unsigned int *exceptionDetail4*) `[static]`

7.4.3.161 static __forceinline__ __device__ void optixThrowException (

int *exceptionCode,*
unsigned int *exceptionDetail0,*
unsigned int *exceptionDetail1,*
unsigned int *exceptionDetail2,*
unsigned int *exceptionDetail3,*
unsigned int *exceptionDetail4,*
unsigned int *exceptionDetail5*) `[static]`

7.4.3.162 static __forceinline__ __device__ void optixThrowException (

int *exceptionCode,*
unsigned int *exceptionDetail0,*
unsigned int *exceptionDetail1,*
unsigned int *exceptionDetail2,*
unsigned int *exceptionDetail3,*
unsigned int *exceptionDetail4,*
unsigned int *exceptionDetail5,*
unsigned int *exceptionDetail6*) `[static]`

7.4.3.163 static __forceinline__ __device__ void optixThrowException (

int *exceptionCode,*
unsigned int *exceptionDetail0,*
unsigned int *exceptionDetail1,*
unsigned int *exceptionDetail2,*
unsigned int *exceptionDetail3,*
unsigned int *exceptionDetail4,*
unsigned int *exceptionDetail5,*
unsigned int *exceptionDetail6,*
unsigned int *exceptionDetail7*) `[static]`

7.4.3.164 template<typename... Payload> static __forceinline__ __device__ void optixTrace (

OptixTraversableHandle *handle,*
float3 *rayOrigin,*
float3 *rayDirection,*
float *tmin,*
float *tmax,*

float *rayTime,*
OptixVisibilityMask *visibilityMask,*
unsigned int *rayFlags,*
unsigned int *SBToffset,*
unsigned int *SBTstride,*
unsigned int *missSBTIndex,*
Payload &... *payload* **)** `[static]`

7.4.3.165 template<typename... Payload> static __forceinline__ __device__ void optixTrace (

OptixPayloadTypeID *type,*
OptixTraversableHandle *handle,*
float3 *rayOrigin,*
float3 *rayDirection,*
float *tmin,*
float *tmax,*
float *rayTime,*
OptixVisibilityMask *visibilityMask,*
unsigned int *rayFlags,*
unsigned int *SBToffset,*
unsigned int *SBTstride,*
unsigned int *missSBTIndex,*
Payload &... *payload* **)** `[static]`

7.4.3.166 static __forceinline__ __device__ float3 optixTransformNormalFromObject-ToWorldSpace (

float3 *normal* **)** `[static]`

7.4.3.167 static __forceinline__ __device__ float3 optixTransformNormalFromWorldToOb-jectSpace (

float3 *normal* **)** `[static]`

7.4.3.168 static __forceinline__ __device__ float3 optixTransformPointFromObject-ToWorldSpace (

float3 *point* **)** `[static]`

7.4.3.169 static __forceinline__ __device__ float3 optixTransformPointFromWorldToOb-jectSpace (

float3 *point* **)** `[static]`

7.4.3.170 static __forceinline__ __device__ float3 optixTransformVectorFromObject-ToWorldSpace (

float3 *vec*) `[static]`

7.4.3.171 static __forceinline__ __device__ float3 optixTransformVectorFromWorldToObjectSpace (

float3 *vec*) `[static]`

7.4.3.172 static __forceinline__ __device__ unsigned int optixUndefinedValue ( ) `[static]`

7.5 optix_7_device_impl_exception.h File Reference

Namespaces

- optix_impl

Constant Groups

- optix_impl

Functions

- static __forceinline__ __device__ void optix_impl::optixDumpStaticTransformFromHandle (OptixTraversableHandle handle)
- static __forceinline__ __device__ void optix_impl::optixDumpMotionMatrixTransformFromHandle (OptixTraversableHandle handle)
- static __forceinline__ __device__ void optix_impl::optixDumpSrtMatrixTransformFromHandle (OptixTraversableHandle handle)
- static __forceinline__ __device__ void optix_impl::optixDumpInstanceFromHandle (OptixTraversableHandle handle)
- static __forceinline__ __device__ void optix_impl::optixDumpTransform (OptixTraversableHandle handle)
- static __forceinline__ __device__ void optix_impl::optixDumpTransformList ()
- static __forceinline__ __device__ void optix_impl::optixDumpExceptionDetails ()

7.5.1 Detailed Description

OptiX public API.

Author

NVIDIA Corporation OptiX public API Reference - Device side implementation for exception helper function.

7.6 optix_7_device_impl_transformations.h File Reference

Namespaces

- optix_impl

Constant Groups

- optix_impl

Functions

- static __forceinline__
  __device__ float4 optix_impl::optixAddFloat4 (const float4 &a, const float4 &b)
- static __forceinline__
  __device__ float4 optix_impl::optixMulFloat4 (const float4 &a, float b)
- static __forceinline__
  __device__ uint4 optix_impl::optixLdg (unsigned long long addr)
- template<class T >
  static __forceinline__ __device__ T optix_impl::optixLoadReadOnlyAlign16 (const T ∗ptr)
- static __forceinline__
  __device__ float4 optix_impl::optixMultiplyRowMatrix (const float4 vec, const float4 m0, const float4 m1, const float4 m2)
- static __forceinline__
  __device__ void optix_impl::optixGetMatrixFromSrt (float4 &m0, float4 &m1, float4 &m2, const OptixSRTData &srt)
- static __forceinline__
  __device__ void optix_impl::optixInvertMatrix (float4 &m0, float4 &m1, float4 &m2)
- static __forceinline__
  __device__ void optix_impl::optixLoadInterpolatedMatrixKey (float4 &m0, float4 &m1, float4 &m2, const float4 ∗matrix, const float t1)
- static __forceinline__
  __device__ void optix_impl::optixLoadInterpolatedSrtKey (float4 &srt0, float4 &srt1, float4 &srt2, float4 &srt3, const float4 ∗srt, const float t1)
- static __forceinline__
  __device__ void optix_impl::optixResolveMotionKey (float &localt, int &key, const OptixMotionOptions &options, const float globalt)
- static __forceinline__
  __device__ void optix_impl::optixGetInterpolatedTransformation (float4 &trf0, float4 &trf1, float4 &trf2, const OptixMatrixMotionTransform ∗transformData, const float time)
- static __forceinline__
  __device__ void optix_impl::optixGetInterpolatedTransformation (float4 &trf0, float4 &trf1, float4 &trf2, const OptixSRTMotionTransform ∗transformData, const float time)
- static __forceinline__
  __device__ void optix_impl::optixGetInterpolatedTransformationFromHandle (float4 &trf0, float4 &trf1, float4 &trf2, const OptixTraversableHandle handle, const float time, const bool objectToWorld)

- static __forceinline__
  __device__ void optix_impl::optixGetWorldToObjectTransformMatrix (float4 &m0, float4 &m1, float4 &m2)
- static __forceinline__
  __device__ void optix_impl::optixGetObjectToWorldTransformMatrix (float4 &m0, float4 &m1, float4 &m2)
- static __forceinline__
  __device__ float3 optix_impl::optixTransformPoint (const float4 &m0, const float4 &m1, const float4 &m2, const float3 &p)
- static __forceinline__
  __device__ float3 optix_impl::optixTransformVector (const float4 &m0, const float4 &m1, const float4 &m2, const float3 &v)
- static __forceinline__
  __device__ float3 optix_impl::optixTransformNormal (const float4 &m0, const float4 &m1, const float4 &m2, const float3 &n)

7.6.1 Detailed Description

OptiX public API.

Author

NVIDIA Corporation OptiX public API Reference - Device side implementation for transformation helper functions.

7.7 optix_7_host.h File Reference

Functions

- const char ∗ optixGetErrorName (OptixResult result)
- const char ∗ optixGetErrorString (OptixResult result)
- OptixResult optixDeviceContextCreate (CUcontext fromContext, const OptixDeviceContextOptions ∗options, OptixDeviceContext ∗context)
- OptixResult optixDeviceContextDestroy (OptixDeviceContext context)
- OptixResult optixDeviceContextGetProperty (OptixDeviceContext context, OptixDeviceProperty property, void ∗value, size_t sizeInBytes)
- OptixResult optixDeviceContextSetLogCallback (OptixDeviceContext context, OptixLogCallback callbackFunction, void ∗callbackData, unsigned int callbackLevel)
- OptixResult optixDeviceContextSetCacheEnabled (OptixDeviceContext context, int enabled)
- OptixResult optixDeviceContextSetCacheLocation (OptixDeviceContext context, const char ∗location)
- OptixResult optixDeviceContextSetCacheDatabaseSizes (OptixDeviceContext context, size_t lowWaterMark, size_t highWaterMark)
- OptixResult optixDeviceContextGetCacheEnabled (OptixDeviceContext context, int ∗enabled)

- OptixResult optixDeviceContextGetCacheLocation (OptixDeviceContext context, char ∗location, size_t locationSize)
- OptixResult optixDeviceContextGetCacheDatabaseSizes (OptixDeviceContext context, size_t ∗lowWaterMark, size_t ∗highWaterMark)
- OptixResult optixPipelineCreate (OptixDeviceContext context, const OptixPipelineCompileOptions ∗pipelineCompileOptions, const OptixPipelineLinkOptions ∗pipelineLinkOptions, const OptixProgramGroup ∗programGroups, unsigned int numProgramGroups, char ∗logString, size_t ∗logStringSize, OptixPipeline ∗pipeline)
- OptixResult optixPipelineDestroy (OptixPipeline pipeline)
- OptixResult optixPipelineSetStackSize (OptixPipeline pipeline, unsigned int directCallableStackSizeFromTraversal, unsigned int directCallableStackSizeFromState, unsigned int continuationStackSize, unsigned int maxTraversableGraphDepth)
- OptixResult optixModuleCreateFromPTX (OptixDeviceContext context, const OptixModuleCompileOptions ∗moduleCompileOptions, const OptixPipelineCompileOptions ∗pipelineCompileOptions, const char ∗PTX, size_t PTXsize, char ∗logString, size_t ∗logStringSize, OptixModule ∗module)
- OptixResult optixModuleCreateFromPTXWithTasks (OptixDeviceContext context, const OptixModuleCompileOptions ∗moduleCompileOptions, const OptixPipelineCompileOptions ∗pipelineCompileOptions, const char ∗PTX, size_t PTXsize, char ∗logString, size_t ∗logStringSize, OptixModule ∗module, OptixTask ∗firstTask)
- OptixResult optixModuleGetCompilationState (OptixModule module, OptixModuleCompileState ∗state)
- OptixResult optixModuleDestroy (OptixModule module)
- OptixResult optixBuiltinISModuleGet (OptixDeviceContext context, const OptixModuleCompileOptions ∗moduleCompileOptions, const OptixPipelineCompileOptions ∗pipelineCompileOptions, const OptixBuiltinISOptions ∗builtinISOptions, OptixModule ∗builtinModule)
- OptixResult optixTaskExecute (OptixTask task, OptixTask ∗additionalTasks, unsigned int maxNumAdditionalTasks, unsigned int ∗numAdditionalTasksCreated)
- OptixResult optixProgramGroupGetStackSize (OptixProgramGroup programGroup, OptixStackSizes ∗stackSizes)
- OptixResult optixProgramGroupCreate (OptixDeviceContext context, const OptixProgramGroupDesc ∗programDescriptions, unsigned int numProgramGroups, const OptixProgramGroupOptions ∗options, char ∗logString, size_t ∗logStringSize, OptixProgramGroup ∗programGroups)
- OptixResult optixProgramGroupDestroy (OptixProgramGroup programGroup)
- OptixResult optixLaunch (OptixPipeline pipeline, CUstream stream, CUdeviceptr pipelineParams, size_t pipelineParamsSize, const OptixShaderBindingTable ∗sbt, unsigned int width, unsigned int height, unsigned int depth)
- OptixResult optixSbtRecordPackHeader (OptixProgramGroup programGroup, void ∗sbtRecordHeaderHostPointer)
- OptixResult optixAccelComputeMemoryUsage (OptixDeviceContext context, const OptixAccelBuildOptions ∗accelOptions, const OptixBuildInput ∗buildInputs, unsigned int numBuildInputs, OptixAccelBufferSizes ∗bufferSizes)

- OptixResult optixAccelBuild (OptixDeviceContext context, CUstream stream, const OptixAccelBuildOptions ∗accelOptions, const OptixBuildInput ∗buildInputs, unsigned int numBuildInputs, CUdeviceptr tempBuffer, size_t tempBufferSizeInBytes, CUdeviceptr outputBuffer, size_t outputBufferSizeInBytes, OptixTraversableHandle ∗outputHandle, const OptixAccelEmitDesc ∗emittedProperties, unsigned int numEmittedProperties)
- OptixResult optixAccelGetRelocationInfo (OptixDeviceContext context, OptixTraversableHandle handle, OptixAccelRelocationInfo ∗info)
- OptixResult optixAccelCheckRelocationCompatibility (OptixDeviceContext context, const OptixAccelRelocationInfo ∗info, int ∗compatible)
- OptixResult optixAccelRelocate (OptixDeviceContext context, CUstream stream, const OptixAccelRelocationInfo ∗info, CUdeviceptr instanceTraversableHandles, size_t numInstanceTraversableHandles, CUdeviceptr targetAccel, size_t targetAccelSizeInBytes, OptixTraversableHandle ∗targetHandle)
- OptixResult optixAccelCompact (OptixDeviceContext context, CUstream stream, OptixTraversableHandle inputHandle, CUdeviceptr outputBuffer, size_t outputBufferSizeInBytes, OptixTraversableHandle ∗outputHandle)
- OptixResult optixConvertPointerToTraversableHandle (OptixDeviceContext onDevice, CUdeviceptr pointer, OptixTraversableType traversableType, OptixTraversableHandle ∗traversableHandle)
- OptixResult optixDenoiserCreate (OptixDeviceContext context, OptixDenoiserModelKind modelKind, const OptixDenoiserOptions ∗options, OptixDenoiser ∗denoiser)
- OptixResult optixDenoiserCreateWithUserModel (OptixDeviceContext context, const void ∗userData, size_t userDataSizeInBytes, OptixDenoiser ∗denoiser)
- OptixResult optixDenoiserDestroy (OptixDenoiser denoiser)
- OptixResult optixDenoiserComputeMemoryResources (const OptixDenoiser denoiser, unsigned int outputWidth, unsigned int outputHeight, OptixDenoiserSizes ∗returnSizes)
- OptixResult optixDenoiserSetup (OptixDenoiser denoiser, CUstream stream, unsigned int inputWidth, unsigned int inputHeight, CUdeviceptr denoiserState, size_t denoiserStateSizeInBytes, CUdeviceptr scratch, size_t scratchSizeInBytes)
- OptixResult optixDenoiserInvoke (OptixDenoiser denoiser, CUstream stream, const OptixDenoiserParams ∗params, CUdeviceptr denoiserState, size_t denoiserStateSizeInBytes, const OptixDenoiserGuideLayer ∗guideLayer, const OptixDenoiserLayer ∗layers, unsigned int numLayers, unsigned int inputOffsetX, unsigned int inputOffsetY, CUdeviceptr scratch, size_t scratchSizeInBytes)
- OptixResult optixDenoiserComputeIntensity (OptixDenoiser denoiser, CUstream stream, const OptixImage2D ∗inputImage, CUdeviceptr outputIntensity, CUdeviceptr scratch, size_t scratchSizeInBytes)
- OptixResult optixDenoiserComputeAverageColor (OptixDenoiser denoiser, CUstream stream, const OptixImage2D ∗inputImage, CUdeviceptr outputAverageColor, CUdeviceptr scratch, size_t scratchSizeInBytes)

7.7.1 Detailed Description

OptiX public API header.

Author

NVIDIA Corporation OptiX host include file – includes the host api if compiling host code. For the math library routines include optix_math.h

7.8 optix_7_types.h File Reference

Classes

- struct OptixDeviceContextOptions
- struct OptixBuildInputTriangleArray
- struct OptixBuildInputCurveArray
- struct OptixBuildInputSphereArray
- struct OptixAabb
- struct OptixBuildInputCustomPrimitiveArray
- struct OptixBuildInputInstanceArray
- struct OptixBuildInput
- struct OptixInstance
- struct OptixMotionOptions
- struct OptixAccelBuildOptions
- struct OptixAccelBufferSizes
- struct OptixAccelEmitDesc
- struct OptixAccelRelocationInfo
- struct OptixStaticTransform
- struct OptixMatrixMotionTransform
- struct OptixSRTData
- struct OptixSRTMotionTransform
- struct OptixImage2D
- struct OptixDenoiserOptions
- struct OptixDenoiserGuideLayer
- struct OptixDenoiserLayer
- struct OptixDenoiserParams
- struct OptixDenoiserSizes
- struct OptixModuleCompileBoundValueEntry
- struct OptixPayloadType
- struct OptixModuleCompileOptions
- struct OptixProgramGroupSingleModule
- struct OptixProgramGroupHitgroup
- struct OptixProgramGroupCallables
- struct OptixProgramGroupDesc
- struct OptixProgramGroupOptions
- struct OptixPipelineCompileOptions

- struct OptixPipelineLinkOptions
- struct OptixShaderBindingTable
- struct OptixStackSizes
- struct OptixBuiltinISOptions

Macros

- #define OPTIX_SBT_RECORD_HEADER_SIZE ((size_t)32)
- #define OPTIX_SBT_RECORD_ALIGNMENT 16ull
- #define OPTIX_ACCEL_BUFFER_BYTE_ALIGNMENT 128ull
- #define OPTIX_INSTANCE_BYTE_ALIGNMENT 16ull
- #define OPTIX_AABB_BUFFER_BYTE_ALIGNMENT 8ull
- #define OPTIX_GEOMETRY_TRANSFORM_BYTE_ALIGNMENT 16ull
- #define OPTIX_TRANSFORM_BYTE_ALIGNMENT 64ull
- #define OPTIX_COMPILE_DEFAULT_MAX_REGISTER_COUNT 0
- #define OPTIX_COMPILE_DEFAULT_MAX_PAYLOAD_TYPE_COUNT 8
- #define OPTIX_COMPILE_DEFAULT_MAX_PAYLOAD_VALUE_COUNT 32

Typedefs

- typedef unsigned long long CUdeviceptr
- typedef struct
  OptixDeviceContext_t ∗ OptixDeviceContext
- typedef struct OptixModule_t ∗ OptixModule
- typedef struct
  OptixProgramGroup_t ∗ OptixProgramGroup
- typedef struct OptixPipeline_t ∗ OptixPipeline
- typedef struct OptixDenoiser_t ∗ OptixDenoiser
- typedef struct OptixTask_t ∗ OptixTask
- typedef unsigned long long OptixTraversableHandle
- typedef unsigned int OptixVisibilityMask
- typedef enum OptixResult OptixResult
- typedef enum OptixDeviceProperty OptixDeviceProperty
- typedef void(∗ OptixLogCallback)(unsigned int level, const char ∗tag, const char ∗message, void ∗cbdata)
- typedef enum
  OptixDeviceContextValidationMode OptixDeviceContextValidationMode
- typedef struct
  OptixDeviceContextOptions OptixDeviceContextOptions
- typedef enum OptixGeometryFlags OptixGeometryFlags
- typedef enum OptixHitKind OptixHitKind
- typedef enum OptixIndicesFormat OptixIndicesFormat
- typedef enum OptixVertexFormat OptixVertexFormat

- typedef enum OptixTransformFormat OptixTransformFormat
- typedef struct
  OptixBuildInputTriangleArray OptixBuildInputTriangleArray
- typedef enum OptixPrimitiveType OptixPrimitiveType
- typedef enum
  OptixPrimitiveTypeFlags OptixPrimitiveTypeFlags
- typedef enum OptixCurveEndcapFlags OptixCurveEndcapFlags
- typedef struct
  OptixBuildInputCurveArray OptixBuildInputCurveArray
- typedef struct
  OptixBuildInputSphereArray OptixBuildInputSphereArray
- typedef struct OptixAabb OptixAabb
- typedef struct
  OptixBuildInputCustomPrimitiveArray OptixBuildInputCustomPrimitiveArray
- typedef struct
  OptixBuildInputInstanceArray OptixBuildInputInstanceArray
- typedef enum OptixBuildInputType OptixBuildInputType
- typedef struct OptixBuildInput OptixBuildInput
- typedef enum OptixInstanceFlags OptixInstanceFlags
- typedef struct OptixInstance OptixInstance
- typedef enum OptixBuildFlags OptixBuildFlags
- typedef enum OptixBuildOperation OptixBuildOperation
- typedef enum OptixMotionFlags OptixMotionFlags
- typedef struct OptixMotionOptions OptixMotionOptions
- typedef struct
  OptixAccelBuildOptions OptixAccelBuildOptions
- typedef struct
  OptixAccelBufferSizes OptixAccelBufferSizes
- typedef enum OptixAccelPropertyType OptixAccelPropertyType
- typedef struct OptixAccelEmitDesc OptixAccelEmitDesc
- typedef struct
  OptixAccelRelocationInfo OptixAccelRelocationInfo
- typedef struct OptixStaticTransform OptixStaticTransform
- typedef struct
  OptixMatrixMotionTransform OptixMatrixMotionTransform
- typedef struct OptixSRTData OptixSRTData
- typedef struct
  OptixSRTMotionTransform OptixSRTMotionTransform
- typedef enum OptixTraversableType OptixTraversableType
- typedef enum OptixPixelFormat OptixPixelFormat
- typedef struct OptixImage2D OptixImage2D
- typedef enum OptixDenoiserModelKind OptixDenoiserModelKind
- typedef struct OptixDenoiserOptions OptixDenoiserOptions

- typedef struct OptixDenoiserGuideLayer OptixDenoiserGuideLayer
- typedef struct OptixDenoiserLayer OptixDenoiserLayer
- typedef enum OptixDenoiserAlphaMode OptixDenoiserAlphaMode
- typedef struct OptixDenoiserParams OptixDenoiserParams
- typedef struct OptixDenoiserSizes OptixDenoiserSizes
- typedef enum OptixRayFlags OptixRayFlags
- typedef enum OptixTransformType OptixTransformType
- typedef enum OptixTraversableGraphFlags OptixTraversableGraphFlags
- typedef enum OptixCompileOptimizationLevel OptixCompileOptimizationLevel
- typedef enum OptixCompileDebugLevel OptixCompileDebugLevel
- typedef enum OptixModuleCompileState OptixModuleCompileState
- typedef struct OptixModuleCompileBoundValueEntry OptixModuleCompileBoundValueEntry
- typedef enum OptixPayloadTypeID OptixPayloadTypeID
- typedef enum OptixPayloadSemantics OptixPayloadSemantics
- typedef struct OptixPayloadType OptixPayloadType
- typedef struct OptixModuleCompileOptions OptixModuleCompileOptions
- typedef enum OptixProgramGroupKind OptixProgramGroupKind
- typedef enum OptixProgramGroupFlags OptixProgramGroupFlags
- typedef struct OptixProgramGroupSingleModule OptixProgramGroupSingleModule
- typedef struct OptixProgramGroupHitgroup OptixProgramGroupHitgroup
- typedef struct OptixProgramGroupCallables OptixProgramGroupCallables
- typedef struct OptixProgramGroupDesc OptixProgramGroupDesc
- typedef struct OptixProgramGroupOptions OptixProgramGroupOptions
- typedef enum OptixExceptionCodes OptixExceptionCodes
- typedef enum OptixExceptionFlags OptixExceptionFlags
- typedef struct OptixPipelineCompileOptions OptixPipelineCompileOptions
- typedef struct OptixPipelineLinkOptions OptixPipelineLinkOptions
- typedef struct OptixShaderBindingTable OptixShaderBindingTable
- typedef struct OptixStackSizes OptixStackSizes

- typedef enum
  OptixQueryFunctionTableOptions OptixQueryFunctionTableOptions
- typedef OptixResult(OptixQueryFunctionTable_t)(int abiId, unsigned int numOptions,
  OptixQueryFunctionTableOptions ∗, const void ∗∗, void ∗functionTable, size_t sizeOfTable)
- typedef struct
  OptixBuiltinISOptions OptixBuiltinISOptions

Enumerations

- enum OptixResult {
  OPTIX_SUCCESS = 0,
  OPTIX_ERROR_INVALID_VALUE = 7001,
  OPTIX_ERROR_HOST_OUT_OF_MEMORY = 7002,
  OPTIX_ERROR_INVALID_OPERATION = 7003,
  OPTIX_ERROR_FILE_IO_ERROR = 7004,
  OPTIX_ERROR_INVALID_FILE_FORMAT = 7005,
  OPTIX_ERROR_DISK_CACHE_INVALID_PATH = 7010,
  OPTIX_ERROR_DISK_CACHE_PERMISSION_ERROR = 7011,
  OPTIX_ERROR_DISK_CACHE_DATABASE_ERROR = 7012,
  OPTIX_ERROR_DISK_CACHE_INVALID_DATA = 7013,
  OPTIX_ERROR_LAUNCH_FAILURE = 7050,
  OPTIX_ERROR_INVALID_DEVICE_CONTEXT = 7051,
  OPTIX_ERROR_CUDA_NOT_INITIALIZED = 7052,
  OPTIX_ERROR_VALIDATION_FAILURE = 7053,
  OPTIX_ERROR_INVALID_PTX = 7200,
  OPTIX_ERROR_INVALID_LAUNCH_PARAMETER = 7201,
  OPTIX_ERROR_INVALID_PAYLOAD_ACCESS = 7202,
  OPTIX_ERROR_INVALID_ATTRIBUTE_ACCESS = 7203,
  OPTIX_ERROR_INVALID_FUNCTION_USE = 7204,
  OPTIX_ERROR_INVALID_FUNCTION_ARGUMENTS = 7205,
  OPTIX_ERROR_PIPELINE_OUT_OF_CONSTANT_MEMORY = 7250,
  OPTIX_ERROR_PIPELINE_LINK_ERROR = 7251,
  OPTIX_ERROR_ILLEGAL_DURING_TASK_EXECUTE = 7270,
  OPTIX_ERROR_INTERNAL_COMPILER_ERROR = 7299,
  OPTIX_ERROR_DENOISER_MODEL_NOT_SET = 7300,
  OPTIX_ERROR_DENOISER_NOT_INITIALIZED = 7301,
  OPTIX_ERROR_ACCEL_NOT_COMPATIBLE = 7400,
  OPTIX_ERROR_PAYLOAD_TYPE_MISMATCH = 7500,
  OPTIX_ERROR_PAYLOAD_TYPE_RESOLUTION_FAILED = 7501,
  OPTIX_ERROR_PAYLOAD_TYPE_ID_INVALID = 7502,
  OPTIX_ERROR_NOT_SUPPORTED = 7800,
  OPTIX_ERROR_UNSUPPORTED_ABI_VERSION = 7801,
  OPTIX_ERROR_FUNCTION_TABLE_SIZE_MISMATCH = 7802,
  OPTIX_ERROR_INVALID_ENTRY_FUNCTION_OPTIONS = 7803,
  OPTIX_ERROR_LIBRARY_NOT_FOUND = 7804,
  OPTIX_ERROR_ENTRY_SYMBOL_NOT_FOUND = 7805,
  OPTIX_ERROR_LIBRARY_UNLOAD_FAILURE = 7806,
  OPTIX_ERROR_DEVICE_OUT_OF_MEMORY = 7807,

OPTIX_ERROR_CUDA_ERROR = 7900,
OPTIX_ERROR_INTERNAL_ERROR = 7990,
OPTIX_ERROR_UNKNOWN = 7999 }

- enum OptixDeviceProperty {
  OPTIX_DEVICE_PROPERTY_LIMIT_MAX_TRACE_DEPTH = 0x2001,
  OPTIX_DEVICE_PROPERTY_LIMIT_MAX_TRAVERSABLE_GRAPH_DEPTH = 0x2002,
  OPTIX_DEVICE_PROPERTY_LIMIT_MAX_PRIMITIVES_PER_GAS = 0x2003,
  OPTIX_DEVICE_PROPERTY_LIMIT_MAX_INSTANCES_PER_IAS = 0x2004,
  OPTIX_DEVICE_PROPERTY_RTCORE_VERSION = 0x2005,
  OPTIX_DEVICE_PROPERTY_LIMIT_MAX_INSTANCE_ID = 0x2006,
  OPTIX_DEVICE_PROPERTY_LIMIT_NUM_BITS_INSTANCE_VISIBILITY_MASK = 0x2007,
  OPTIX_DEVICE_PROPERTY_LIMIT_MAX_SBT_RECORDS_PER_GAS = 0x2008,
  OPTIX_DEVICE_PROPERTY_LIMIT_MAX_SBT_OFFSET = 0x2009 }
- enum OptixDeviceContextValidationMode {
  OPTIX_DEVICE_CONTEXT_VALIDATION_MODE_OFF = 0,
  OPTIX_DEVICE_CONTEXT_VALIDATION_MODE_ALL = 0xFFFFFFFF }
- enum OptixGeometryFlags {
  OPTIX_GEOMETRY_FLAG_NONE = 0,
  OPTIX_GEOMETRY_FLAG_DISABLE_ANYHIT = 1u << 0,
  OPTIX_GEOMETRY_FLAG_REQUIRE_SINGLE_ANYHIT_CALL = 1u << 1,
  OPTIX_GEOMETRY_FLAG_DISABLE_TRIANGLE_FACE_CULLING = 1u << 2 }
- enum OptixHitKind {
  OPTIX_HIT_KIND_TRIANGLE_FRONT_FACE = 0xFE,
  OPTIX_HIT_KIND_TRIANGLE_BACK_FACE = 0xFF }
- enum OptixIndicesFormat {
  OPTIX_INDICES_FORMAT_NONE = 0,
  OPTIX_INDICES_FORMAT_UNSIGNED_SHORT3 = 0x2102,
  OPTIX_INDICES_FORMAT_UNSIGNED_INT3 = 0x2103 }
- enum OptixVertexFormat {
  OPTIX_VERTEX_FORMAT_NONE = 0,
  OPTIX_VERTEX_FORMAT_FLOAT3 = 0x2121,
  OPTIX_VERTEX_FORMAT_FLOAT2 = 0x2122,
  OPTIX_VERTEX_FORMAT_HALF3 = 0x2123,
  OPTIX_VERTEX_FORMAT_HALF2 = 0x2124,
  OPTIX_VERTEX_FORMAT_SNORM16_3 = 0x2125,
  OPTIX_VERTEX_FORMAT_SNORM16_2 = 0x2126 }
- enum OptixTransformFormat {
  OPTIX_TRANSFORM_FORMAT_NONE = 0,
  OPTIX_TRANSFORM_FORMAT_MATRIX_FLOAT12 = 0x21E1 }
- enum OptixPrimitiveType {
  OPTIX_PRIMITIVE_TYPE_CUSTOM = 0x2500,
  OPTIX_PRIMITIVE_TYPE_ROUND_QUADRATIC_BSPLINE = 0x2501,
  OPTIX_PRIMITIVE_TYPE_ROUND_CUBIC_BSPLINE = 0x2502,
  OPTIX_PRIMITIVE_TYPE_ROUND_LINEAR = 0x2503,
  OPTIX_PRIMITIVE_TYPE_ROUND_CATMULLROM = 0x2504,
  OPTIX_PRIMITIVE_TYPE_SPHERE = 0x2506,
  OPTIX_PRIMITIVE_TYPE_TRIANGLE = 0x2531 }

- enum OptixPrimitiveTypeFlags {
  OPTIX_PRIMITIVE_TYPE_FLAGS_CUSTOM = 1 << 0,
  OPTIX_PRIMITIVE_TYPE_FLAGS_ROUND_QUADRATIC_BSPLINE = 1 << 1,
  OPTIX_PRIMITIVE_TYPE_FLAGS_ROUND_CUBIC_BSPLINE = 1 << 2,
  OPTIX_PRIMITIVE_TYPE_FLAGS_ROUND_LINEAR = 1 << 3,
  OPTIX_PRIMITIVE_TYPE_FLAGS_ROUND_CATMULLROM = 1 << 4,
  OPTIX_PRIMITIVE_TYPE_FLAGS_SPHERE = 1 << 6,
  OPTIX_PRIMITIVE_TYPE_FLAGS_TRIANGLE = 1 << 31 }
- enum OptixCurveEndcapFlags {
  OPTIX_CURVE_ENDCAP_DEFAULT = 0,
  OPTIX_CURVE_ENDCAP_ON = 1 << 0 }
- enum OptixBuildInputType {
  OPTIX_BUILD_INPUT_TYPE_TRIANGLES = 0x2141,
  OPTIX_BUILD_INPUT_TYPE_CUSTOM_PRIMITIVES = 0x2142,
  OPTIX_BUILD_INPUT_TYPE_INSTANCES = 0x2143,
  OPTIX_BUILD_INPUT_TYPE_INSTANCE_POINTERS = 0x2144,
  OPTIX_BUILD_INPUT_TYPE_CURVES = 0x2145,
  OPTIX_BUILD_INPUT_TYPE_SPHERES = 0x2146 }
- enum OptixInstanceFlags {
  OPTIX_INSTANCE_FLAG_NONE = 0,
  OPTIX_INSTANCE_FLAG_DISABLE_TRIANGLE_FACE_CULLING = 1u << 0,
  OPTIX_INSTANCE_FLAG_FLIP_TRIANGLE_FACING = 1u << 1,
  OPTIX_INSTANCE_FLAG_DISABLE_ANYHIT = 1u << 2,
  OPTIX_INSTANCE_FLAG_ENFORCE_ANYHIT = 1u << 3 }
- enum OptixBuildFlags {
  OPTIX_BUILD_FLAG_NONE = 0,
  OPTIX_BUILD_FLAG_ALLOW_UPDATE = 1u << 0,
  OPTIX_BUILD_FLAG_ALLOW_COMPACTION = 1u << 1,
  OPTIX_BUILD_FLAG_PREFER_FAST_TRACE = 1u << 2,
  OPTIX_BUILD_FLAG_PREFER_FAST_BUILD = 1u << 3,
  OPTIX_BUILD_FLAG_ALLOW_RANDOM_VERTEX_ACCESS = 1u << 4,
  OPTIX_BUILD_FLAG_ALLOW_RANDOM_INSTANCE_ACCESS = 1u << 5 }
- enum OptixBuildOperation {
  OPTIX_BUILD_OPERATION_BUILD = 0x2161,
  OPTIX_BUILD_OPERATION_UPDATE = 0x2162 }
- enum OptixMotionFlags {
  OPTIX_MOTION_FLAG_NONE = 0,
  OPTIX_MOTION_FLAG_START_VANISH = 1u << 0,
  OPTIX_MOTION_FLAG_END_VANISH = 1u << 1 }
- enum OptixAccelPropertyType {
  OPTIX_PROPERTY_TYPE_COMPACTED_SIZE = 0x2181,
  OPTIX_PROPERTY_TYPE_AABBS = 0x2182 }
- enum OptixTraversableType {
  OPTIX_TRAVERSABLE_TYPE_STATIC_TRANSFORM = 0x21C1,
  OPTIX_TRAVERSABLE_TYPE_MATRIX_MOTION_TRANSFORM = 0x21C2,
  OPTIX_TRAVERSABLE_TYPE_SRT_MOTION_TRANSFORM = 0x21C3 }
- enum OptixPixelFormat {

OPTIX_PIXEL_FORMAT_HALF2 = 0x2207,
OPTIX_PIXEL_FORMAT_HALF3 = 0x2201,
OPTIX_PIXEL_FORMAT_HALF4 = 0x2202,
OPTIX_PIXEL_FORMAT_FLOAT2 = 0x2208,
OPTIX_PIXEL_FORMAT_FLOAT3 = 0x2203,
OPTIX_PIXEL_FORMAT_FLOAT4 = 0x2204,
OPTIX_PIXEL_FORMAT_UCHAR3 = 0x2205,
OPTIX_PIXEL_FORMAT_UCHAR4 = 0x2206,
OPTIX_PIXEL_FORMAT_INTERNAL_GUIDE_LAYER = 0x2209 }

- enum OptixDenoiserModelKind {
  OPTIX_DENOISER_MODEL_KIND_LDR = 0x2322,
  OPTIX_DENOISER_MODEL_KIND_HDR = 0x2323,
  OPTIX_DENOISER_MODEL_KIND_AOV = 0x2324,
  OPTIX_DENOISER_MODEL_KIND_TEMPORAL = 0x2325,
  OPTIX_DENOISER_MODEL_KIND_TEMPORAL_AOV = 0x2326,
  OPTIX_DENOISER_MODEL_KIND_UPSCALE2X = 0x2327,
  OPTIX_DENOISER_MODEL_KIND_TEMPORAL_UPSCALE2X = 0x2328 }
- enum OptixDenoiserAlphaMode {
  OPTIX_DENOISER_ALPHA_MODE_COPY = 0,
  OPTIX_DENOISER_ALPHA_MODE_ALPHA_AS_AOV = 1,
  OPTIX_DENOISER_ALPHA_MODE_FULL_DENOISE_PASS = 2 }
- enum OptixRayFlags {
  OPTIX_RAY_FLAG_NONE = 0u,
  OPTIX_RAY_FLAG_DISABLE_ANYHIT = 1u << 0,
  OPTIX_RAY_FLAG_ENFORCE_ANYHIT = 1u << 1,
  OPTIX_RAY_FLAG_TERMINATE_ON_FIRST_HIT = 1u << 2,
  OPTIX_RAY_FLAG_DISABLE_CLOSESTHIT = 1u << 3,
  OPTIX_RAY_FLAG_CULL_BACK_FACING_TRIANGLES = 1u << 4,
  OPTIX_RAY_FLAG_CULL_FRONT_FACING_TRIANGLES = 1u << 5,
  OPTIX_RAY_FLAG_CULL_DISABLED_ANYHIT = 1u << 6,
  OPTIX_RAY_FLAG_CULL_ENFORCED_ANYHIT = 1u << 7 }
- enum OptixTransformType {
  OPTIX_TRANSFORM_TYPE_NONE = 0,
  OPTIX_TRANSFORM_TYPE_STATIC_TRANSFORM = 1,
  OPTIX_TRANSFORM_TYPE_MATRIX_MOTION_TRANSFORM = 2,
  OPTIX_TRANSFORM_TYPE_SRT_MOTION_TRANSFORM = 3,
  OPTIX_TRANSFORM_TYPE_INSTANCE = 4 }
- enum OptixTraversableGraphFlags {
  OPTIX_TRAVERSABLE_GRAPH_FLAG_ALLOW_ANY = 0,
  OPTIX_TRAVERSABLE_GRAPH_FLAG_ALLOW_SINGLE_GAS = 1u << 0,
  OPTIX_TRAVERSABLE_GRAPH_FLAG_ALLOW_SINGLE_LEVEL_INSTANCING = 1u << 1 }
- enum OptixCompileOptimizationLevel {
  OPTIX_COMPILE_OPTIMIZATION_DEFAULT = 0,
  OPTIX_COMPILE_OPTIMIZATION_LEVEL_0 = 0x2340,
  OPTIX_COMPILE_OPTIMIZATION_LEVEL_1 = 0x2341,
  OPTIX_COMPILE_OPTIMIZATION_LEVEL_2 = 0x2342,
  OPTIX_COMPILE_OPTIMIZATION_LEVEL_3 = 0x2343 }
- enum OptixCompileDebugLevel {

OPTIX_COMPILE_DEBUG_LEVEL_DEFAULT = 0,
OPTIX_COMPILE_DEBUG_LEVEL_NONE = 0x2350,
OPTIX_COMPILE_DEBUG_LEVEL_MINIMAL = 0x2351,
OPTIX_COMPILE_DEBUG_LEVEL_MODERATE = 0x2353,
OPTIX_COMPILE_DEBUG_LEVEL_FULL = 0x2352 }

- enum OptixModuleCompileState {
  OPTIX_MODULE_COMPILE_STATE_NOT_STARTED = 0x2360,
  OPTIX_MODULE_COMPILE_STATE_STARTED = 0x2361,
  OPTIX_MODULE_COMPILE_STATE_IMPENDING_FAILURE = 0x2362,
  OPTIX_MODULE_COMPILE_STATE_FAILED = 0x2363,
  OPTIX_MODULE_COMPILE_STATE_COMPLETED = 0x2364 }
- enum OptixPayloadTypeID {
  OPTIX_PAYLOAD_TYPE_DEFAULT = 0,
  OPTIX_PAYLOAD_TYPE_ID_0 = (1 << 0u),
  OPTIX_PAYLOAD_TYPE_ID_1 = (1 << 1u),
  OPTIX_PAYLOAD_TYPE_ID_2 = (1 << 2u),
  OPTIX_PAYLOAD_TYPE_ID_3 = (1 << 3u),
  OPTIX_PAYLOAD_TYPE_ID_4 = (1 << 4u),
  OPTIX_PAYLOAD_TYPE_ID_5 = (1 << 5u),
  OPTIX_PAYLOAD_TYPE_ID_6 = (1 << 6u),
  OPTIX_PAYLOAD_TYPE_ID_7 = (1 << 7u) }
- enum OptixPayloadSemantics {
  OPTIX_PAYLOAD_SEMANTICS_TRACE_CALLER_NONE = 0,
  OPTIX_PAYLOAD_SEMANTICS_TRACE_CALLER_READ = 1u << 0,
  OPTIX_PAYLOAD_SEMANTICS_TRACE_CALLER_WRITE = 2u << 0,
  OPTIX_PAYLOAD_SEMANTICS_TRACE_CALLER_READ_WRITE = 3u << 0,
  OPTIX_PAYLOAD_SEMANTICS_CH_NONE = 0,
  OPTIX_PAYLOAD_SEMANTICS_CH_READ = 1u << 2,
  OPTIX_PAYLOAD_SEMANTICS_CH_WRITE = 2u << 2,
  OPTIX_PAYLOAD_SEMANTICS_CH_READ_WRITE = 3u << 2,
  OPTIX_PAYLOAD_SEMANTICS_MS_NONE = 0,
  OPTIX_PAYLOAD_SEMANTICS_MS_READ = 1u << 4,
  OPTIX_PAYLOAD_SEMANTICS_MS_WRITE = 2u << 4,
  OPTIX_PAYLOAD_SEMANTICS_MS_READ_WRITE = 3u << 4,
  OPTIX_PAYLOAD_SEMANTICS_AH_NONE = 0,
  OPTIX_PAYLOAD_SEMANTICS_AH_READ = 1u << 6,
  OPTIX_PAYLOAD_SEMANTICS_AH_WRITE = 2u << 6,
  OPTIX_PAYLOAD_SEMANTICS_AH_READ_WRITE = 3u << 6,
  OPTIX_PAYLOAD_SEMANTICS_IS_NONE = 0,
  OPTIX_PAYLOAD_SEMANTICS_IS_READ = 1u << 8,
  OPTIX_PAYLOAD_SEMANTICS_IS_WRITE = 2u << 8,
  OPTIX_PAYLOAD_SEMANTICS_IS_READ_WRITE = 3u << 8 }
- enum OptixProgramGroupKind {
  OPTIX_PROGRAM_GROUP_KIND_RAYGEN = 0x2421,
  OPTIX_PROGRAM_GROUP_KIND_MISS = 0x2422,
  OPTIX_PROGRAM_GROUP_KIND_EXCEPTION = 0x2423,
  OPTIX_PROGRAM_GROUP_KIND_HITGROUP = 0x2424,
  OPTIX_PROGRAM_GROUP_KIND_CALLABLES = 0x2425 }

- enum OptixProgramGroupFlags { OPTIX_PROGRAM_GROUP_FLAGS_NONE = 0 }
- enum OptixExceptionCodes {
  OPTIX_EXCEPTION_CODE_STACK_OVERFLOW = -1,
  OPTIX_EXCEPTION_CODE_TRACE_DEPTH_EXCEEDED = -2,
  OPTIX_EXCEPTION_CODE_TRAVERSAL_DEPTH_EXCEEDED = -3,
  OPTIX_EXCEPTION_CODE_TRAVERSAL_INVALID_TRAVERSABLE = -5,
  OPTIX_EXCEPTION_CODE_TRAVERSAL_INVALID_MISS_SBT = -6,
  OPTIX_EXCEPTION_CODE_TRAVERSAL_INVALID_HIT_SBT = -7,
  OPTIX_EXCEPTION_CODE_UNSUPPORTED_PRIMITIVE_TYPE = -8,
  OPTIX_EXCEPTION_CODE_INVALID_RAY = -9,
  OPTIX_EXCEPTION_CODE_CALLABLE_PARAMETER_MISMATCH = -10,
  OPTIX_EXCEPTION_CODE_BUILTIN_IS_MISMATCH = -11,
  OPTIX_EXCEPTION_CODE_CALLABLE_INVALID_SBT = -12,
  OPTIX_EXCEPTION_CODE_CALLABLE_NO_DC_SBT_RECORD = -13,
  OPTIX_EXCEPTION_CODE_CALLABLE_NO_CC_SBT_RECORD = -14,
  OPTIX_EXCEPTION_CODE_UNSUPPORTED_SINGLE_LEVEL_GAS = -15,
  OPTIX_EXCEPTION_CODE_INVALID_VALUE_ARGUMENT_0 = -16,
  OPTIX_EXCEPTION_CODE_INVALID_VALUE_ARGUMENT_1 = -17,
  OPTIX_EXCEPTION_CODE_INVALID_VALUE_ARGUMENT_2 = -18,
  OPTIX_EXCEPTION_CODE_UNSUPPORTED_DATA_ACCESS = -32,
  OPTIX_EXCEPTION_CODE_PAYLOAD_TYPE_MISMATCH = -33 }
- enum OptixExceptionFlags {
  OPTIX_EXCEPTION_FLAG_NONE = 0,
  OPTIX_EXCEPTION_FLAG_STACK_OVERFLOW = 1u << 0,
  OPTIX_EXCEPTION_FLAG_TRACE_DEPTH = 1u << 1,
  OPTIX_EXCEPTION_FLAG_USER = 1u << 2,
  OPTIX_EXCEPTION_FLAG_DEBUG = 1u << 3 }
- enum OptixQueryFunctionTableOptions {
  OPTIX_QUERY_FUNCTION_TABLE_OPTION_DUMMY = 0 }

7.8.1 Detailed Description

OptiX public API header.

Author

NVIDIA Corporation OptiX types include file – defines types and enums used by the API. For the math library routines include optix_math.h

7.9 optix_denoiser_tiling.h File Reference

Classes

- struct OptixUtilDenoiserImageTile

Functions

- OptixResult optixUtilGetPixelStride (const OptixImage2D &image, unsigned int &pixelStrideInBytes)
- OptixResult optixUtilDenoiserSplitImage (const OptixImage2D &input, const OptixImage2D &output, unsigned int overlapWindowSizeInPixels, unsigned int tileWidth, unsigned int tileHeight, std::vector< OptixUtilDenoiserImageTile > &tiles)
- OptixResult optixUtilDenoiserInvokeTiled (OptixDenoiser denoiser, CUstream stream, const OptixDenoiserParams ∗params, CUdeviceptr denoiserState, size_t denoiserStateSizeInBytes, const OptixDenoiserGuideLayer ∗guideLayer, const OptixDenoiserLayer ∗layers, unsigned int numLayers, CUdeviceptr scratch, size_t scratchSizeInBytes, unsigned int overlapWindowSizeInPixels, unsigned int tileWidth, unsigned int tileHeight)

7.9.1 Detailed Description

OptiX public API header.

Author

NVIDIA Corporation

7.10 optix_device.h File Reference

Macros

- #define __UNDEF_OPTIX_INCLUDE_INTERNAL_HEADERS_OPTIX_DEVICE_H__

7.10.1 Detailed Description

OptiX public API.

Author

NVIDIA Corporation OptiX public API Reference - Host/Device side

7.10.2 Macro Definition Documentation

7.10.2.1 #define __UNDEF_OPTIX_INCLUDE_INTERNAL_HEADERS_OPTIX_DEVICE_H__

7.11 optix_function_table.h File Reference

Classes

- struct OptixFunctionTable

Macros

- #define OPTIX_ABI_VERSION 61

Typedefs

- typedef struct OptixFunctionTable OptixFunctionTable

7.11.1 Detailed Description

OptiX public API header.

Author

NVIDIA Corporation

7.11.2 Macro Definition Documentation

7.11.2.1 #define OPTIX_ABI_VERSION 61

The OptiX ABI version.

7.12 optix_function_table_definition.h File Reference

Variables

- OptixFunctionTable g_optixFunctionTable

7.12.1 Detailed Description

OptiX public API header.

Author

NVIDIA Corporation

7.13 optix_host.h File Reference

Macros

- #define __UNDEF_OPTIX_INCLUDE_INTERNAL_HEADERS_OPTIX_HOST_H__

7.13.1 Detailed Description

OptiX public API.

Author

NVIDIA Corporation OptiX public API Reference - Host side

7.13.2 Macro Definition Documentation

7.13.2.1 #define __UNDEF_OPTIX_INCLUDE_INTERNAL_HEADERS_OPTIX_HOST_H__

7.14 optix_stack_size.h File Reference

Functions

- OptixResult optixUtilAccumulateStackSizes (OptixProgramGroup programGroup, OptixStackSizes ∗stackSizes)
- OptixResult optixUtilComputeStackSizes (const OptixStackSizes ∗stackSizes, unsigned int maxTraceDepth, unsigned int maxCCDepth, unsigned int maxDCDepth, unsigned int ∗directCallableStackSizeFromTraversal, unsigned int ∗directCallableStackSizeFromState, unsigned int ∗continuationStackSize)
- OptixResult optixUtilComputeStackSizesDCSplit (const OptixStackSizes ∗stackSizes, unsigned int dssDCFromTraversal, unsigned int dssDCFromState, unsigned int maxTraceDepth, unsigned int maxCCDepth, unsigned int maxDCDepthFromTraversal, unsigned int maxDCDepthFromState, unsigned int ∗directCallableStackSizeFromTraversal, unsigned int ∗directCallableStackSizeFromState, unsigned int ∗continuationStackSize)
- OptixResult optixUtilComputeStackSizesCssCCTree (const OptixStackSizes ∗stackSizes, unsigned int cssCCTree, unsigned int maxTraceDepth, unsigned int maxDCDepth, unsigned int ∗directCallableStackSizeFromTraversal, unsigned int ∗directCallableStackSizeFromState, unsigned int ∗continuationStackSize)
- OptixResult optixUtilComputeStackSizesSimplePathTracer (OptixProgramGroup programGroupRG, OptixProgramGroup programGroupMS1, const OptixProgramGroup ∗programGroupCH1, unsigned int programGroupCH1Count, OptixProgramGroup programGroupMS2, const OptixProgramGroup ∗programGroupCH2, unsigned int programGroupCH2Count, unsigned int ∗directCallableStackSizeFromTraversal, unsigned int ∗directCallableStackSizeFromState, unsigned int ∗continuationStackSize)

7.14.1 Detailed Description

OptiX public API header.

Author

NVIDIA Corporation

7.15 optix_stubs.h File Reference

Macros

- #define WIN32_LEAN_AND_MEAN 1

Functions

- static void ∗ optixLoadWindowsDllFromName (const char ∗optixDllName)

- static void ∗ optixLoadWindowsDll ()
- OptixResult optixInitWithHandle (void ∗∗handlePtr)
- OptixResult optixInit (void)
- OptixResult optixUninitWithHandle (void ∗handle)

Variables

- OptixFunctionTable g_optixFunctionTable

7.15.1 Detailed Description

OptiX public API header.

Author

NVIDIA Corporation

7.15.2 Macro Definition Documentation

7.15.2.1 #define WIN32_LEAN_AND_MEAN 1

7.15.3 Function Documentation

7.15.3.1 static void∗ optixLoadWindowsDll ( ) `[static]`

7.15.3.2 static void∗ optixLoadWindowsDllFromName (const char ∗ *optixDllName*) `[static]`

7.16 optix_types.h File Reference

Macros

- #define __OPTIX_INCLUDE_INTERNAL_HEADERS__
- #define __UNDEF_OPTIX_INCLUDE_INTERNAL_HEADERS_OPTIX_TYPES_H__

7.16.1 Detailed Description

OptiX public API header.

Author

NVIDIA Corporation

7.16.2 Macro Definition Documentation

7.16.2.1 #define __OPTIX_INCLUDE_INTERNAL_HEADERS__

7.16.2.2 #define __UNDEF_OPTIX_INCLUDE_INTERNAL_HEADERS_OPTIX_TYPES_H__