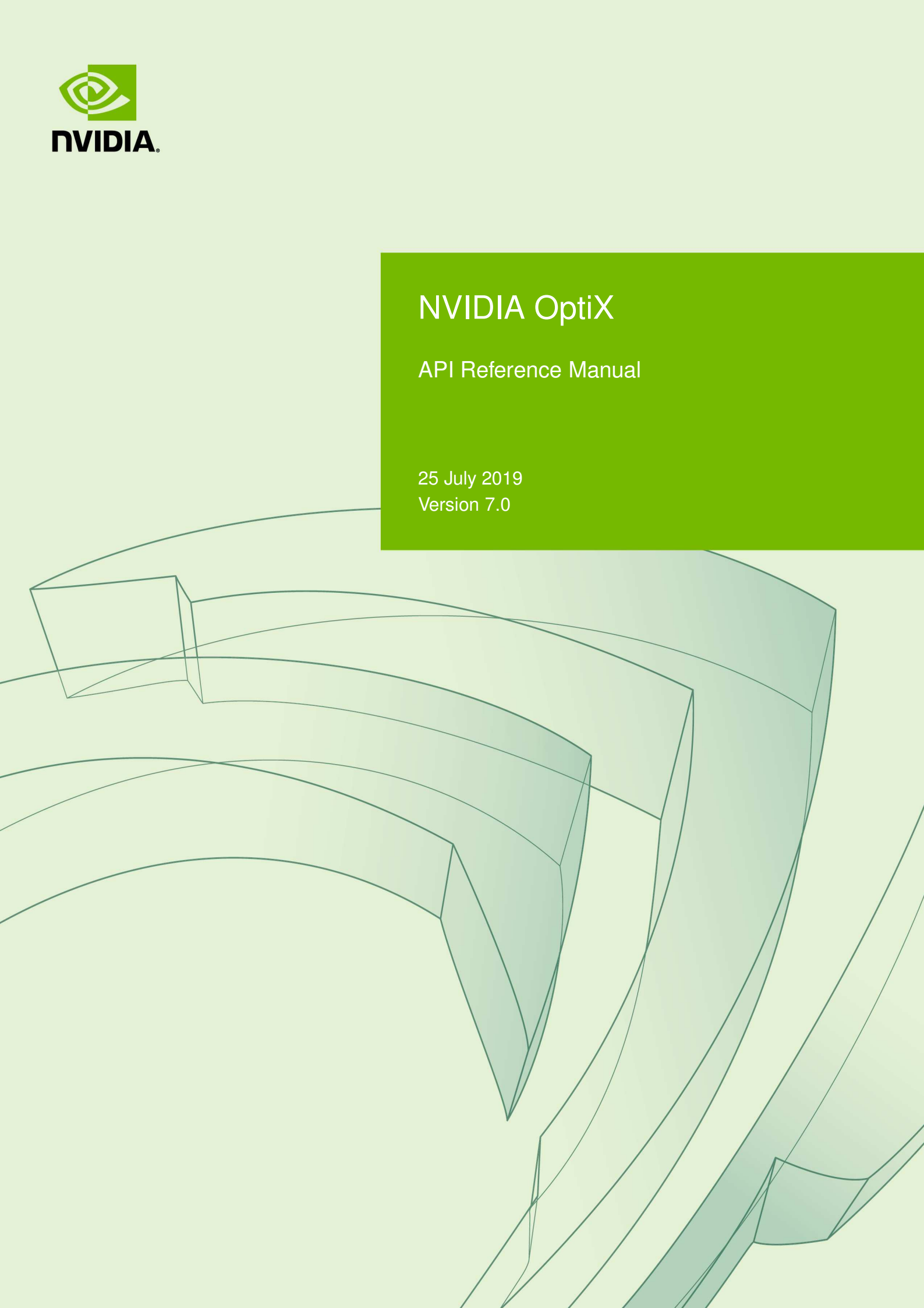

NVIDIA OptiX

API Reference Manual

25 July 2019
Version 7.0

Contents

1 Module Index

1.1 Modules

Here is a list of all modules:

2 Class Index

2.1 Class List

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

3 Module Documentation

3.1 Device API

Functions

- 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)
- 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, unsigned int &p0)
- 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, unsigned int &p0, unsigned int &p1)
- 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, unsigned int &p0, unsigned int &p1, unsigned int &p2)

- 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, unsigned int &p0, unsigned int &p1, unsigned int &p2, unsigned int &p3)
- 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, unsigned int &p0, unsigned int &p1, unsigned int &p2, unsigned int &p3, unsigned int &p4)
- 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, unsigned int &p0, unsigned int &p1, unsigned int &p2, unsigned int &p3, unsigned int &p4, unsigned int &p5)
- 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, unsigned int &p0, unsigned int &p1, unsigned int &p2, unsigned int &p3, unsigned int &p4, unsigned int &p5, unsigned int &p6)
- 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, unsigned int &p0, unsigned int &p1, unsigned int &p2, unsigned int &p3, unsigned int &p4, unsigned int &p5, unsigned int &p6, unsigned int &p7)
- 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__ 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 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__ void optixGetTriangleVertexData (OptixTraversableHandle gas, unsigned int primIdx, unsigned int sbtGASIndex, float time, float3 data[3])
- 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__ 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 optixGetInstanceId ()
- static __forceinline__ __device__ unsigned int optixGetInstanceIndex ()
- static __forceinline__ __device__ unsigned int optixGetHitKind ()

- static __forceinline__ __device__ bool optixIsTriangleHit ()
- static __forceinline__ __device__ bool optixIsTriangleFrontFaceHit ()
- static __forceinline__ __device__ bool optixIsTriangleBackFaceHit ()
- 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 ()

3.1.1 Detailed Description

OptiX Device API.

3.1.2 Function Documentation

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

Returns the attribute at slot 0.

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

Returns the attribute at slot 1.

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

Returns the attribute at slot 2.

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

Returns the attribute at slot 3.

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

Returns the attribute at slot 4.

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

Returns the attribute at slot 5.

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

Returns the attribute at slot 6.

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

Returns the attribute at slot 7.

3.1.2.9 static __forceinline__ __device__ int optixGetExceptionCode ( ) [static]

Returns the exception code.

Only available in EX.

3.1.2.10 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.

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

Returns the 32-bit exception detail at slot 1.

See Also

optixGetExceptionDetail_0()

3.1.2.12 static __forceinline__ __device__ unsigned int optixGetExceptionDetail_2 ( ) [static]

Returns the 32-bit exception detail at slot 2.

See Also

optixGetExceptionDetail_0()

3.1.2.13 static __forceinline__ __device__ unsigned int optixGetExceptionDetail_3 ( ) [static]

Returns the 32-bit exception detail at slot 3.

See Also

optixGetExceptionDetail_0()

3.1.2.14 static __forceinline__ __device__ unsigned int optixGetExceptionDetail_4 ( ) [static]

Returns the 32-bit exception detail at slot 4.

See Also

optixGetExceptionDetail_0()

3.1.2.15 static __forceinline__ __device__ unsigned int optixGetExceptionDetail_5 ( ) [static]

Returns the 32-bit exception detail at slot 5.

See Also

optixGetExceptionDetail_0()

3.1.2.16 static __forceinline__ __device__ unsigned int optixGetExceptionDetail_6 ( ) [static]

Returns the 32-bit exception detail at slot 6.

See Also

optixGetExceptionDetail_0()

3.1.2.17 static __forceinline__ __device__ unsigned int optixGetExceptionDetail_7 ( ) [static]

Returns the 32-bit exception detail at slot 7.

See Also

optixGetExceptionDetail_0()

3.1.2.18 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.

3.1.2.19 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.

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

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

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

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

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

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

3.1.2.23 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.

3.1.2.24 static __forceinline__ __device__ unsigned int optixGetHitKind ( ) [static]

AH - Returns the 8 bit hit kind associated with the current optixReportIntersection. For custom intersection, this will be the hitKind passed to optixReportIntersection.

For built-in intersection see table below. IS, CH - returns the hitKind from the last recorded intersection. For custom intersection, this will be the hitKind passed to optixReportIntersection. For built-in intersection see table below.

```
Built in intersection hitKind values:
   result & 0x80 : built-in primitive hit
   result == 0xFE: front face of triangle hit
   result == 0xFF: back face of triangle hit
```

3.1.2.25 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

3.1.2.26 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.

3.1.2.27 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, optixGetInstanceId returns 0

3.1.2.28 static __forceinline__ __device__ const float4* optixGetInstanceInverseTransformFromHandle (

OptixTraversableHandle *handle*) [static]

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

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

3.1.2.29 static __forceinline__ __device__ const float4* optixGetInstanceTransformFromHandle (

OptixTraversableHandle *handle*) [static]

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

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

3.1.2.30 static __forceinline__ __device__ uint3 optixGetLaunchDimensions ( ) [static]

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

3.1.2.31 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.

3.1.2.32 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.

3.1.2.33 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.

3.1.2.34 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.

3.1.2.35 static __forceinline__ __device__ void optixGetObjectToWorldTransformMatrix (float *m[12]*) [static]

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

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

3.1.2.36 static __forceinline__ __device__ unsigned int optixGetPayload_0 ( ) [static]

Reads the 32-bit payload value at slot 0.

3.1.2.37 static __forceinline__ __device__ unsigned int optixGetPayload_1 ( ) [static]

Reads the 32-bit payload value at slot 1.

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

Reads the 32-bit payload value at slot 2.

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

Reads the 32-bit payload value at slot 3.

3.1.2.40 static __forceinline__ __device__ unsigned int optixGetPayload_4 ( ) [static]

Reads the 32-bit payload value at slot 4.

3.1.2.41 static __forceinline__ __device__ unsigned int optixGetPayload_5 ( ) [static]

Reads the 32-bit payload value at slot 5.

3.1.2.42 static __forceinline__ __device__ unsigned int optixGetPayload_6 ( ) [static]

Reads the 32-bit payload value at slot 6.

3.1.2.43 static __forceinline__ __device__ unsigned int optixGetPayload_7 ( ) [static]

Reads the 32-bit payload value at slot 7.

3.1.2.44 static __forceinline__ __device__ unsigned int optixGetPrimitiveIndex ( ) [static]

For a given OptixBuildInputTriangleArray the number of primitives is defined as (OptixBuildInputTriangleArray::indexBuffer == nullptr) ? OptixBuildInputTriangleArray::numVertices/3 : OptixBuildInputTriangleArray::numIndices/3;

For a given OptixBuildInputCustomPrimitiveArray the number of primitives is defined as numAabbs. The primitive index returns is the index into the corresponding build array 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. In EX with exception code OPTIX_EXCEPTION_CODE_TRAVERSAL_INVALID_HIT_SBT corresponds to the active primitive index. Returns zero for all other exceptions.

3.1.2.45 static __forceinline__ __device__ unsigned int optixGetRayFlags ( ) [static]

Returns the rayFlags passed into rtTrace

Only available in IS, AH, CH, MS

3.1.2.46 static __forceinline__ __device__ float optixGetRayTime ( ) [static]

Returns the rayTime passed into rtTrace.

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

3.1.2.47 static __forceinline__ __device__ float optixGetRayTmax ( ) [static]

In IS and CH returns the current smallest reported hitT or the tmax passed into rtTrace 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

3.1.2.48 static __forceinline__ __device__ float optixGetRayTmin ( ) [static]

Returns the tmin passed into rtTrace.

Only available in IS, AH, CH, MS

3.1.2.49 static __forceinline__ __device__ unsigned int optixGetRayVisibilityMask ( ) [static]

Returns the visibilityMask passed into rtTrace

Only available in IS, AH, CH, MS

3.1.2.50 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.

3.1.2.51 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.

3.1.2.52 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.

3.1.2.53 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, MS, EX

3.1.2.54 static __forceinline__ __device__ unsigned int optixGetTransformListSize ( ) [static]

Returns the number of transforms on the current transform list.

Only available in IS, AH, CH, MS, EX

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

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

3.1.2.56 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.

3.1.2.57 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.

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

3.1.2.58 static __forceinline__ __device__ float3 optixGetWorldRayDirection ( ) [static]

Returns the rayDirection passed into rtTrace.

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

3.1.2.59 static __forceinline__ __device__ float3 optixGetWorldRayOrigin ( ) [static]

Returns the rayOrigin passed into rtTrace.

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

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

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

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

3.1.2.61 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.

3.1.2.62 static __forceinline__ __device__ bool optixIsTriangleBackFaceHit ( ) [static]

Convenience function interpreting the result of optixGetHitKind().

3.1.2.63 static __forceinline__ __device__ bool optixIsTriangleFrontFaceHit ( ) [static]

Convenience function interpreting the result of optixGetHitKind().

3.1.2.64 static __forceinline__ __device__ bool optixIsTriangleHit ( ) [static]

Convenience function interpreting the result of optixGetHitKind().

3.1.2.65 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*	

3.1.2.66 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)

3.1.2.67 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)

3.1.2.68 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)

3.1.2.69 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)

3.1.2.70 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)

3.1.2.71 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)

3.1.2.72 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)

3.1.2.73 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)

3.1.2.74 static __forceinline__ __device__ void optixSetPayload_0 (

unsigned int *p*) `[static]`

Writes the 32-bit payload value at slot 0.

3.1.2.75 static __forceinline__ __device__ void optixSetPayload_1 (

unsigned int *p*) `[static]`

Writes the 32-bit payload value at slot 1.

3.1.2.76 static __forceinline__ __device__ void optixSetPayload_2 (

unsigned int *p*) `[static]`

Writes the 32-bit payload value at slot 2.

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

Writes the 32-bit payload value at slot 3.

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

Writes the 32-bit payload value at slot 4.

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

Writes the 32-bit payload value at slot 5.

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

Writes the 32-bit payload value at slot 6.

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

Writes the 32-bit payload value at slot 7.

3.1.2.82 static __forceinline__ __device__ void optixTerminateRay ( ) [static]

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

Available only in AH.

3.1.2.83 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.

3.1.2.84 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)

3.1.2.85 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)

3.1.2.86 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)

3.1.2.87 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)

3.1.2.88 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)

3.1.2.89 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)

3.1.2.90 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)

3.1.2.91 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)

3.1.2.92 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*) `[static]`

Initiates a ray tracing query starting with the given traversable (overload without payload).

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

3.1.2.93 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,***
unsigned int & ***p0*** **)** `[static]`

Initiates a ray tracing query starting with the given traversable (overload with 1 payload register).

See Also

optixTrace(OptixTraversableHandle,float3,float3,float,float,float,OptixVisibilityMask,unsigned int,unsigned int,unsig

3.1.2.94 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,***
unsigned int & ***p0,***
unsigned int & ***p1*** **)** `[static]`

Initiates a ray tracing query starting with the given traversable (overload with 2 payload registers).

See Also

optixTrace(OptixTraversableHandle,float3,float3,float,float,float,OptixVisibilityMask,unsigned int,unsigned int,unsig

3.1.2.95 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,*
unsigned int & *p0,*
unsigned int & *p1,*
unsigned int & *p2*) `[static]`

Initiates a ray tracing query starting with the given traversable (overload with 3 payload registers).

See Also

optixTrace(OptixTraversableHandle,float3,float3,float,float,float,OptixVisibilityMask,unsigned int,unsigned int,unsig

3.1.2.96 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,*
unsigned int & *p0,*
unsigned int & *p1,*
unsigned int & *p2,*
unsigned int & *p3*) `[static]`

Initiates a ray tracing query starting with the given traversable (overload with 4 payload registers).

See Also

optixTrace(OptixTraversableHandle,float3,float3,float,float,float,OptixVisibilityMask,unsigned int,unsigned int,unsig

3.1.2.97 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,***
unsigned int & ***p0,***
unsigned int & ***p1,***
unsigned int & ***p2,***
unsigned int & ***p3,***
unsigned int & ***p4*** **)** `[static]`

Initiates a ray tracing query starting with the given traversable (overload with 5 payload registers).

See Also

optixTrace(OptixTraversableHandle,float3,float3,float,float,float,OptixVisibilityMask,unsigned int,unsigned int,unsig

3.1.2.98 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,***
unsigned int & ***p0,***
unsigned int & ***p1,***
unsigned int & ***p2,***
unsigned int & ***p3,***
unsigned int & ***p4,***
unsigned int & ***p5*** **)** `[static]`

Initiates a ray tracing query starting with the given traversable (overload with 6 payload registers).

See Also

optixTrace(OptixTraversableHandle,float3,float3,float,float,float,OptixVisibilityMask,unsigned int,unsigned int,unsig

3.1.2.99 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,*
unsigned int & *p0,*
unsigned int & *p1,*
unsigned int & *p2,*
unsigned int & *p3,*
unsigned int & *p4,*
unsigned int & *p5,*
unsigned int & *p6*) `[static]`

Initiates a ray tracing query starting with the given traversable (overload with 7 payload registers).

See Also

optixTrace(OptixTraversableHandle,float3,float3,float,float,float,OptixVisibilityMask,unsigned int,unsigned int,unsig

3.1.2.100 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,*
unsigned int & *p0,*

unsigned int & *p1*,
unsigned int & *p2*,
unsigned int & *p3*,
unsigned int & *p4*,
unsigned int & *p5*,
unsigned int & *p6*,
unsigned int & *p7*) [static]

Initiates a ray tracing query starting with the given traversable (overload with 8 payload registers).

See Also

optixTrace(OptixTraversableHandle,float3,float3,float,float,float,OptixVisibilityMask,unsigned int,unsigned int,unsig

3.1.2.101 static __forceinline__ __device__ float3 optixTransformNormalFromObjectToWorldSpace (

float3 *normal*) [static]

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

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

3.1.2.102 static __forceinline__ __device__ float3 optixTransformNormalFromWorldToObjectSpace (

float3 *normal*) [static]

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

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

3.1.2.103 static __forceinline__ __device__ float3 optixTransformPointFromObjectToWorldSpace (

float3 *point*) [static]

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

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

3.1.2.104 static __forceinline__ __device__ float3 optixTransformPointFromWorldToObjectSpace (

float3 *point*) [static]

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

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

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

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

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

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

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

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

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

Returns an undefined value.

3.2 Host API

Modules

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

3.2.1 Detailed Description

OptiX Host API.

3.3 Error handling

Functions

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

3.3.1 Detailed Description

3.3.2 Function Documentation

3.3.2.1 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

3.3.2.2 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

3.4 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)

3.4.1 Detailed Description

3.4.2 Function Documentation

3.4.2.1 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

3.4.2.2 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.

3.4.2.3 OptixResult optixDeviceContextGetCacheDatabaseSizes (OptixDeviceContext *context,* size_t * *lowWaterMark,* size_t * *highWaterMark*)

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

Parameters

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

3.4.2.4 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

3.4.2.5 OptixResult optixDeviceContextGetCacheLocation (OptixDeviceContext *context,* char * *location,* size_t *locationSize*)

Returns the location of the disk cache.

Parameters

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

3.4.2.6 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

3.4.2.7 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.

Parameters

in	*context*	the device context
in	*lowWaterMark*	the low water mark

Parameters

in	*highWaterMark*	the high water mark

3.4.2.8 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.

Parameters

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

3.4.2.9 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

3.4.2.10 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

3.5 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)

3.5.1 Detailed Description

3.5.2 Function Documentation

3.5.2.1 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*	

3.5.2.2 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.

3.5.2.3 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.

3.6 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 optixModuleDestroy (OptixModule module)

3.6.1 Detailed Description

3.6.2 Function Documentation

3.6.2.1 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*	
in	*PTXsize*	
out	*logString*	Information will be written to this string. If logStringSize $>$ 0 logString will be null terminated.

Parameters

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.

3.6.2.2 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.

3.7 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)

3.7.1 Detailed Description

3.7.2 Function Documentation

3.7.2.1 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*	

3.7.2.2 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.

3.7.2.3 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

3.8 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)

3.8.1 Detailed Description

3.8.2 Function Documentation

3.8.2.1 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.

3.8.2.2 OptixResult optixSbtRecordPackHeader (OptixProgramGroup *programGroup,* void * *sbtRecordHeaderHostPointer*)

Parameters

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

3.9 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)

3.9.1 Detailed Description

3.9.2 Function Documentation

3.9.2.1 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*	
out	*emittedProperties*	types of requested properties and output buffers OPTIX_PROPERTY_TYPE_AABBS is currently not supported and must be requested via optixAccelEmitProperties
in	*numEmittedProperties*	number of post-build properties to populate (may be zero)

3.9.2.2 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.

3.9.2.3 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*	

3.9.2.4 OptixResult optixAccelComputeMemoryUsage (

OptixDeviceContext *context,*
const OptixAccelBuildOptions * *accelOptions,*
const OptixBuildInput * *buildInputs,*
unsigned int *numBuildInputs,*
OptixAccelBufferSizes * *bufferSizes*)

Parameters

in	*context*	device context of the pipeline
in	*accelOptions*	accel options
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

3.9.2.5 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.

3.9.2.6 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*	

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

Parameters

in	*onDevice*	
in	*pointer*	pointer to traversalbe 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

3.10 Denoiser

Functions

- OptixResult optixDenoiserCreate (OptixDeviceContext context, const OptixDenoiserOptions *options, OptixDenoiser *denoiser)
- OptixResult optixDenoiserSetModel (OptixDenoiser denoiser, OptixDenoiserModelKind kind, void *data, size_t sizeInBytes)
- 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 outputWidth, unsigned int outputHeight, 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 OptixImage2D *inputLayers, unsigned int numInputLayers, unsigned int inputOffsetX, unsigned int inputOffsetY, const OptixImage2D *outputLayer, CUdeviceptr scratch, size_t scratchSizeInBytes)
- OptixResult optixDenoiserComputeIntensity (OptixDenoiser denoiser, CUstream stream, const OptixImage2D *inputImage, CUdeviceptr outputIntensity, CUdeviceptr scratch, size_t scratchSizeInBytes)

3.10.1 Detailed Description

3.10.2 Function Documentation

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

Parameters

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

3.10.2.2 OptixResult optixDenoiserComputeMemoryResources (

const OptixDenoiser *denoiser,*
unsigned int *outputWidth,*
unsigned int *outputHeight,*
OptixDenoiserSizes * *returnSizes*)

Computes the GPU memory resources required to execute the denoiser.

Parameters

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

3.10.2.3 OptixResult optixDenoiserCreate (

OptixDeviceContext *context,*
const OptixDenoiserOptions * *options,*
OptixDenoiser * *denoiser*)

Creates a denoiser object with the given options.

Parameters

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

3.10.2.4 OptixResult optixDenoiserDestroy (

OptixDenoiser *denoiser*)

Destroys the denoiser object and any associated host resources.

3.10.2.5 OptixResult optixDenoiserInvoke (

OptixDenoiser *denoiser,*
CUstream *stream,*
const OptixDenoiserParams * *params,*
CUdeviceptr *denoiserState,*
size_t *denoiserStateSizeInBytes,*
const OptixImage2D * *inputLayers,*
unsigned int *numInputLayers,*
unsigned int *inputOffsetX,*
unsigned int *inputOffsetY,*
const OptixImage2D * *outputLayer,*

CUdeviceptr *scratch,*
size_t *scratchSizeInBytes*)

Invokes denoiser on a set of input data and produces one output image. Scratch memory must be available during the execution of the denoiser.

Parameters

in	*denoiser*	
in	*stream*	
in	*params*	
in	*denoiserState*	
in	*denoiserStateSizeInBytes*	
in	*inputLayers*	
in	*numInputLayers*	
in	*inputOffsetX*	
in	*inputOffsetY*	
in	*outputLayer*	
in	*scratch*	
in	*scratchSizeInBytes*	

3.10.2.6 OptixResult optixDenoiserSetModel (

OptixDenoiser *denoiser,*
OptixDenoiserModelKind *kind,*
void * *data,*
size_t *sizeInBytes*)

Sets the model of the denoiser.

If the kind is OPTIX_DENOISER_MODEL_KIND_USER, then the data and sizeInByes must not be null and zero respectively. For other kinds, these parameters must be zero.

Parameters

in	*denoiser*	
in	*kind*	
in	*data*	
in	*sizeInBytes*	

3.10.2.7 OptixResult optixDenoiserSetup (

OptixDenoiser *denoiser,*
CUstream *stream,*
unsigned int *outputWidth,*

unsigned int *outputHeight,*
CUdeviceptr *denoiserState,*
size_t *denoiserStateSizeInBytes,*
CUdeviceptr *scratch,*
size_t *scratchSizeInBytes*)

Initializes the state required by the denoiser.

Parameters

in	*denoiser*	
in	*stream*	
in	*outputWidth*	
in	*outputHeight*	
in	*denoiserState*	
in	*denoiserStateSizeInBytes*	
in	*scratch*	
in	*scratchSizeInBytes*	

3.11 Types

Classes

- struct OptixDeviceContextOptions
- struct OptixBuildInputTriangleArray
- 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 OptixDenoiserParams
- struct OptixDenoiserSizes
- struct OptixModuleCompileOptions
- struct OptixProgramGroupSingleModule
- struct OptixProgramGroupHitgroup
- struct OptixProgramGroupCallables
- struct OptixProgramGroupDesc
- struct OptixProgramGroupOptions
- struct OptixPipelineCompileOptions
- struct OptixPipelineLinkOptions
- struct OptixShaderBindingTable
- struct OptixStackSizes

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

Typedefs

- typedef unsigned int 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 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 struct OptixDeviceContextOptions OptixDeviceContextOptions
- typedef enum OptixGeometryFlags OptixGeometryFlags
- typedef enum OptixHitKind OptixHitKind
- typedef enum OptixIndicesFormat OptixIndicesFormat
- typedef enum OptixVertexFormat OptixVertexFormat
- typedef struct OptixBuildInputTriangleArray OptixBuildInputTriangleArray
- 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 OptixDenoiserInputKind OptixDenoiserInputKind
- typedef enum OptixDenoiserModelKind OptixDenoiserModelKind
- typedef struct OptixDenoiserOptions OptixDenoiserOptions
- 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 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 ABI_ID, unsigned int numOptions, OptixQueryFunctionTableOptions *, const void **, void *functionTable, size_t sizeOfTable)

Enumerations

- enum OptixResult
- 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 OptixGeometryFlags {
  OPTIX_GEOMETRY_FLAG_NONE = 0,
  OPTIX_GEOMETRY_FLAG_DISABLE_ANYHIT = 1u << 0,
  OPTIX_GEOMETRY_FLAG_REQUIRE_SINGLE_ANYHIT_CALL = 1u << 1 }
- enum OptixHitKind {
  OPTIX_HIT_KIND_TRIANGLE_FRONT_FACE = 0xFE,
  OPTIX_HIT_KIND_TRIANGLE_BACK_FACE = 0xFF }
- enum OptixIndicesFormat {
  OPTIX_INDICES_FORMAT_UNSIGNED_SHORT3 = 0x2102,
  OPTIX_INDICES_FORMAT_UNSIGNED_INT3 = 0x2103 }
- enum OptixVertexFormat {
  OPTIX_VERTEX_FORMAT_FLOAT3 = 0x2121,
  OPTIX_VERTEX_FORMAT_FLOAT2 = 0x2122,
  OPTIX_VERTEX_FORMAT_HALF3 = 0x2123,
  OPTIX_VERTEX_FORMAT_HALF2 = 0x2124 }
- 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 }
- 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,
  OPTIX_INSTANCE_FLAG_DISABLE_TRANSFORM = 1u << 6 }
- enum OptixBuildFlags {
  OPTIX_BUILD_FLAG_NONE = 0,
  OPTIX_BUILD_FLAG_ALLOW_UPDATE = 1u << 0 ,
  OPTIX_BUILD_FLAG_ALLOW_RANDOM_VERTEX_ACCESS = 1u << 4 }
- enum OptixBuildOperation {
  OPTIX_BUILD_OPERATION_BUILD = 0x2161,
  OPTIX_BUILD_OPERATION_UPDATE = 0x2162 }
- enum OptixMotionFlags

- 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_HALF3 = 0x2201,
  OPTIX_PIXEL_FORMAT_HALF4 = 0x2202,
  OPTIX_PIXEL_FORMAT_FLOAT3 = 0x2203,
  OPTIX_PIXEL_FORMAT_FLOAT4 = 0x2204,
  OPTIX_PIXEL_FORMAT_UCHAR3 = 0x2205,
  OPTIX_PIXEL_FORMAT_UCHAR4 = 0x2206 }
- enum OptixDenoiserInputKind
- enum OptixDenoiserModelKind {
  OPTIX_DENOISER_MODEL_KIND_USER = 0x2321,
  OPTIX_DENOISER_MODEL_KIND_LDR = 0x2322,
  OPTIX_DENOISER_MODEL_KIND_HDR = 0x2323 }
- 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_LEVEL_0 = 0,
  OPTIX_COMPILE_OPTIMIZATION_LEVEL_1 = 1,
  OPTIX_COMPILE_OPTIMIZATION_LEVEL_2 = 2,
  OPTIX_COMPILE_OPTIMIZATION_LEVEL_3 = 3 }
- enum OptixCompileDebugLevel {
  OPTIX_COMPILE_DEBUG_LEVEL_NONE = 0,
  OPTIX_COMPILE_DEBUG_LEVEL_LINEINFO = 1,
  OPTIX_COMPILE_DEBUG_LEVEL_FULL = 2 }

- 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 }
- 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 }

3.11.1 Detailed Description

OptiX Types.

3.11.2 Macro Definition Documentation

3.11.2.1 #define OPTIX_AABB_BUFFER_BYTE_ALIGNMENT 8ull

Alignment requirement for OptixBuildInputCustomPrimitiveArray::aabbBuffers and OptixBuildInputInstanceArray::aabbs.

3.11.2.2 #define OPTIX_ACCEL_BUFFER_BYTE_ALIGNMENT 128ull

Alignment requirement for output and temporay buffers for acceleration structures.

3.11.2.3 #define OPTIX_COMPILE_DEFAULT_MAX_REGISTER_COUNT 0

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

3.11.2.4 #define OPTIX_GEOMETRY_TRANSFORM_BYTE_ALIGNMENT 16ull

Alignment requirement for OptixBuildInputTriangleArray::preTransform.

3.11.2.5 #define OPTIX_INSTANCE_BYTE_ALIGNMENT 16ull

Alignment requirement for OptixBuildInputInstanceArray::instances.

3.11.2.6 #define OPTIX_SBT_RECORD_ALIGNMENT 16ull

Alignment requirement for device pointers in OptixShaderBindingTable.

3.11.2.7 #define OPTIX_SBT_RECORD_HEADER_SIZE ((size_t)32)

Size of the SBT record headers.

3.11.2.8 #define OPTIX_TRANSFORM_BYTE_ALIGNMENT 64ull

Alignment requirement for OptixStaticTransform, OptixMatrixMotionTransform, OptixSRTMotionTransform.

3.11.3 Typedef Documentation

3.11.3.1 typedef unsigned int CUdeviceptr

CUDA device pointer.

3.11.3.2 typedef struct OptixAabb OptixAabb

AABB inputs.

3.11.3.3 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()

3.11.3.4 typedef struct OptixAccelBuildOptions OptixAccelBuildOptions

Build options for acceleration structures.

See Also

optixAccelComputeMemoryUsage(), optixAccelBuild()

3.11.3.5 typedef struct OptixAccelEmitDesc OptixAccelEmitDesc

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

See Also

optixAccelBuild()

3.11.3.6 typedef enum OptixAccelPropertyType OptixAccelPropertyType

Properties which can be emitted during acceleration structure build.

See Also

OptixAccelEmitDesc::type.

3.11.3.7 typedef struct OptixAccelRelocationInfo OptixAccelRelocationInfo

Used to store information realated to relocation of acceleration structures.

See Also

optixAccelGetRelocationInfo(), optixAccelCheckRelocationCompatibility(), optixAccelRelocate()

3.11.3.8 typedef enum OptixBuildFlags OptixBuildFlags

Builder Options

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

3.11.3.9 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()

3.11.3.10 typedef struct OptixBuildInputCustomPrimitiveArray OptixBuildInputCustomPrimitiveArray

Custom primitive inputs

See Also

OptixBuildInput::aabbArray

3.11.3.11 typedef struct OptixBuildInputInstanceArray OptixBuildInputInstanceArray

Instance and instance pointer inputs

See Also

OptixBuildInput::instanceArray

3.11.3.12 typedef struct OptixBuildInputTriangleArray OptixBuildInputTriangleArray

Triangle inputs

See Also

OptixBuildInput::triangleArray

3.11.3.13 typedef enum OptixBuildInputType OptixBuildInputType

Enum to distinguish the different build input types.

See Also

OptixBuildInput::type

3.11.3.14 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

3.11.3.15 typedef enum OptixCompileDebugLevel OptixCompileDebugLevel

Debug levels

See Also

OptixModuleCompileOptions::debugLevel

3.11.3.16 typedef enum OptixCompileOptimizationLevel OptixCompileOptimizationLevel

Optimization levels

See Also

OptixModuleCompileOptions::optLevel

3.11.3.17 typedef struct OptixDenoiser_t* OptixDenoiser

Opaque type representing a denoiser instance.

3.11.3.18 typedef enum OptixDenoiserInputKind OptixDenoiserInputKind

Input kinds used by the denoiser.

See Also

OptixDenoiserOptions::inputKind

3.11.3.19 typedef enum OptixDenoiserModelKind OptixDenoiserModelKind

Model kind used by the denoiser.

See Also

optixDenoiserSetModel()

3.11.3.20 typedef struct OptixDenoiserOptions OptixDenoiserOptions

Options used by the denoiser

See Also

optixDenoiserCreate()

3.11.3.21 typedef struct OptixDenoiserParams OptixDenoiserParams

Various parameters used by the denoiser

See Also

optixDenoiserInvoke()

3.11.3.22 typedef struct OptixDenoiserSizes OptixDenoiserSizes

Various sizes related to the denoiser.

See Also

optixDenoiserComputeMemoryResources()

3.11.3.23 typedef struct OptixDeviceContext_t* OptixDeviceContext

Opaque type representing a device context.

3.11.3.24 typedef struct OptixDeviceContextOptions OptixDeviceContextOptions

Parameters used for optixDeviceContextCreate()

See Also

optixDeviceContextCreate()

3.11.3.25 typedef enum OptixDeviceProperty OptixDeviceProperty

Parameters used for optixDeviceContextGetProperty()

See Also

optixDeviceContextGetProperty()

3.11.3.26 typedef enum OptixExceptionCodes OptixExceptionCodes

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

3.11.3.27 typedef enum OptixExceptionFlags OptixExceptionFlags

Exception flags.

See Also

OptixPipelineCompileOptions::exceptionFlags, OptixExceptionCodes

3.11.3.28 typedef enum OptixGeometryFlags OptixGeometryFlags

Flags used by OptixBuildInputTriangleArray::flags and OptixBuildInputCustomPrimitiveArray::flags.

3.11.3.29 typedef enum OptixHitKind OptixHitKind

Hit kind for reporting intersections

See Also

optixReportIntersection()

3.11.3.30 typedef struct OptixImage2D OptixImage2D

Image descriptor used by the denoiser.

See Also

optixDenoiserInvoke(), optixDenoiserComputeIntensity()

3.11.3.31 typedef enum OptixIndicesFormat OptixIndicesFormat

Format of indices used int OptixBuildInputTriangleArray::indexFormat.

3.11.3.32 typedef struct OptixInstance OptixInstance

Instances

See Also

OptixBuildInputInstanceArray::instances

3.11.3.33 typedef enum OptixInstanceFlags OptixInstanceFlags

Flags set on the OptixInstance::flags.

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

3.11.3.34 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

3.11.3.35 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()

3.11.3.36 typedef struct OptixModule_t* OptixModule

Opaque type representing a module.

3.11.3.37 typedef struct OptixModuleCompileOptions OptixModuleCompileOptions

Compilation options for module

See Also

optixModuleCreateFromPTX()

3.11.3.38 typedef enum OptixMotionFlags OptixMotionFlags

Enum to specify motion flags.

See Also

OptixMotionOptions::flags.

3.11.3.39 typedef struct OptixMotionOptions OptixMotionOptions

Motion options

See Also

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

3.11.3.40 typedef struct OptixPipeline_t* OptixPipeline

Opaque type representing a pipeline.

3.11.3.41 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()

3.11.3.42 typedef struct OptixPipelineLinkOptions OptixPipelineLinkOptions

Link options for a pipeline

See Also

optixPipelineCreate()

3.11.3.43 typedef enum OptixPixelFormat OptixPixelFormat

Pixel formats used by the denoiser.

See Also

OptixImage2D::format

3.11.3.44 typedef struct OptixProgramGroup_t* OptixProgramGroup

Opaque type representing a program group.

3.11.3.45 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

3.11.3.46 typedef struct OptixProgramGroupDesc OptixProgramGroupDesc

Descriptor for program groups.

3.11.3.47 typedef enum OptixProgramGroupFlags OptixProgramGroupFlags

Flags for program groups.

3.11.3.48 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

3.11.3.49 typedef enum OptixProgramGroupKind OptixProgramGroupKind

Distinguishes different kinds of program groups.

3.11.3.50 typedef struct OptixProgramGroupOptions OptixProgramGroupOptions

Program group options

See Also

optixProgramGroupCreate()

3.11.3.51 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

3.11.3.52 typedef OptixResult(OptixQueryFunctionTable_t)(int ABI_ID, unsigned int numOptions, OptixQueryFunctionTableOptions *, const void **, void *functionTable, size_t sizeOfTable)

Type of the function `optixQueryFunctionTable()`

3.11.3.53 typedef enum OptixQueryFunctionTableOptions OptixQueryFunctionTableOptions

Options that can be passed to `optixQueryFunctionTable()`

3.11.3.54 typedef enum OptixRayFlags OptixRayFlags

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

See Also

optixTrace()

3.11.3.55 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()

3.11.3.56 typedef struct OptixShaderBindingTable OptixShaderBindingTable

Describes the shader binding table (SBT)

See Also

optixLaunch()

3.11.3.57 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.

```
[  sx   a   b  pvx ]
```

The scaling matrix S = [0 sy c pvy] defines an affine transformation that can include scale, shear, and a [0 0 sz pvz]

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.

```
[  1  0  0 tx ]
```

The translation T = [0 1 0 ty] defines another translation that is applied after the rotation. Typically, this [0 0 1 tz]

translation includes the inverse translation from the matrix S to reverse its effect.

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()

3.11.3.58 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()

3.11.3.59 typedef struct OptixStackSizes OptixStackSizes

Describes the stack size requirements of a program group.

See Also

optixProgramGroupGetStackSize()

3.11.3.60 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()

3.11.3.61 typedef enum OptixTransformType OptixTransformType

Transform

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

3.11.3.62 typedef enum OptixTraversableGraphFlags OptixTraversableGraphFlags

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

3.11.3.63 typedef unsigned long long OptixTraversableHandle

Traversable handle.

3.11.3.64 typedef enum OptixTraversableType OptixTraversableType

Traversable Handles

See Also

optixConvertPointerToTraversableHandle()

3.11.3.65 typedef enum OptixVertexFormat OptixVertexFormat

Format of vertices used in OptixBuildInputTriangleArray::vertexFormat.

3.11.3.66 typedef unsigned int OptixVisibilityMask

Visibility mask.

3.11.4 Enumeration Type Documentation

3.11.4.1 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.

3.11.4.2 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_RANDOM_VERTEX_ACCESS Allow access to random baked vertex in closesthit.

3.11.4.3 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

3.11.4.4 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.

3.11.4.5 enum OptixCompileDebugLevel

Debug levels

See Also

OptixModuleCompileOptions::debugLevel

Enumerator

OPTIX_COMPILE_DEBUG_LEVEL_NONE No debug information.

OPTIX_COMPILE_DEBUG_LEVEL_LINEINFO Generate lineinfo only.

OPTIX_COMPILE_DEBUG_LEVEL_FULL Generate dwarf debug information.

3.11.4.6 enum OptixCompileOptimizationLevel

Optimization levels

See Also

OptixModuleCompileOptions::optLevel

Enumerator

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.

3.11.4.7 enum OptixDenoiserInputKind

Input kinds used by the denoiser.

See Also

OptixDenoiserOptions::inputKind

3.11.4.8 enum OptixDenoiserModelKind

Model kind used by the denoiser.

See Also

optixDenoiserSetModel()

Enumerator

OPTIX_DENOISER_MODEL_KIND_USER Use the model provided by the associated pointer. See the programming guide for a description of how to format the data.

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.

3.11.4.9 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 for the sum of the number of SBT records of all build inputs to a single Geometry Acceleration Structure (GAS). 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)

3.11.4.10 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 Exception details: optixGetExceptionInvalidSbtOffset()

OPTIX_EXCEPTION_CODE_TRAVERSAL_INVALID_HIT_SBT The traversal hit SBT record index out of bounds. Exception details: optixGetTransformListSize() optixGetTransformListHandle() optixGetExceptionInvalidSbtOffset() optixGetPrimitiveIndex()

3.11.4.11 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.

3.11.4.12 enum OptixGeometryFlags

Flags used by OptixBuildInputTriangleArray::flags 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.

3.11.4.13 enum OptixHitKind

Hit kind for reporting intersections

See Also

optixReportIntersection()

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.

3.11.4.14 enum OptixIndicesFormat

Format of indices used int OptixBuildInputTriangleArray::indexFormat.

Enumerator

OPTIX_INDICES_FORMAT_UNSIGNED_SHORT3 Three shorts.

OPTIX_INDICES_FORMAT_UNSIGNED_INT3 Three ints.

3.11.4.15 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.

OPTIX_INSTANCE_FLAG_DISABLE_TRANSFORM Disable the instance transformation.

3.11.4.16 enum OptixMotionFlags

Enum to specify motion flags.

See Also

OptixMotionOptions::flags.

3.11.4.17 enum OptixPixelFormat

Pixel formats used by the denoiser.

See Also

OptixImage2D::format

Enumerator

OPTIX_PIXEL_FORMAT_HALF3 three halfs, RGB

OPTIX_PIXEL_FORMAT_HALF4 four halfs, RGBA

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

3.11.4.18 enum OptixProgramGroupFlags

Flags for program groups.

Enumerator

OPTIX_PROGRAM_GROUP_FLAGS_NONE Currently there are no flags.

3.11.4.19 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 containg 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 containg a direct (DC) or continuation (CC) callable program
See Also

OptixProgramGroupCallables, OptixProgramGroupDesc::callables

3.11.4.20 enum OptixQueryFunctionTableOptions

Options that can be passed to `optixQueryFunctionTable()`

Enumerator

OPTIX_QUERY_FUNCTION_TABLE_OPTION_DUMMY Placeholder (there are no options yet)

3.11.4.21 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 and miss programs for the ray.

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.

3.11.4.22 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()

3.11.4.23 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

3.11.4.24 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.

3.11.4.25 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

3.11.4.26 enum OptixVertexFormat

Format of vertices used in OptixBuildInputTriangleArray::vertexFormat.

Enumerator

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.

3.12 Function Table

Classes

- struct OptixFunctionTable

Typedefs

- typedef struct OptixFunctionTable OptixFunctionTable

Variables

- OptixFunctionTable g_optixFunctionTable

3.12.1 Detailed Description

OptiX Function Table.

3.12.2 Typedef Documentation

3.12.2.1 typedef struct OptixFunctionTable OptixFunctionTable

The function table containing all API functions.

See optixInit() and optixInitWithHandle().

3.12.3 Variable Documentation

3.12.3.1 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.

3.13 Utilities

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 optixInitWithHandle (void **handlePtr)
- OptixResult optixInit (void)

3.13.1 Detailed Description

OptiX Utilities.

3.13.2 Function Documentation

3.13.2.1 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.

3.13.2.2 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.

3.13.2.3 OptixResult optixUtilAccumulateStackSizes (OptixProgramGroup *programGroup*, OptixStackSizes * *stackSizes*)

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.

3.13.2.4 OptixResult optixUtilComputeStackSizes (const OptixStackSizes * *stackSizes*, unsigned int *maxTraceDepth*, unsigned int *maxCCDepth*, unsigned int *maxDCDepth*, unsigned int * *directCallableStackSizeFromTraversal*, unsigned int * *directCallableStackSizeFromState*, unsigned int * *continuationStackSize*)

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.

3.13.2.5 OptixResult optixUtilComputeStackSizesCssCCTree (const OptixStackSizes * *stackSizes*, unsigned int *cssCCTree*, unsigned int *maxTraceDepth*, unsigned int *maxDCDepth*, unsigned int * *directCallableStackSizeFromTraversal*, unsigned int * *directCallableStackSizeFromState*,

unsigned int * *continuationStackSize*)

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.

3.13.2.6 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*)

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.

Parameters

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.

3.13.2.7 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*)

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 Class Documentation

4.1 OptixAabb Struct Reference

Public Attributes

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

4.1.1 Detailed Description

AABB inputs.

4.1.2 Member Data Documentation

4.1.2.1 float OptixAabb::maxX

Upper extent in X direction.

4.1.2.2 float OptixAabb::maxY

Upper extent in Y direction.

4.1.2.3 float OptixAabb::maxZ

Upper extent in Z direction.

4.1.2.4 float OptixAabb::minX

Lower extent in X direction.

4.1.2.5 float OptixAabb::minY

Lower extent in Y direction.

4.1.2.6 float OptixAabb::minZ

Lower extent in Z direction.

4.2 OptixAccelBufferSizes Struct Reference

Public Attributes

- size_t outputSizeInBytes

- size_t tempSizeInBytes
- size_t tempUpdateSizeInBytes

4.2.1 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()

4.2.2 Member Data Documentation

4.2.2.1 size_t OptixAccelBufferSizes::outputSizeInBytes

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

4.2.2.2 size_t OptixAccelBufferSizes::tempSizeInBytes

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

4.2.2.3 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.

4.3 OptixAccelBuildOptions Struct Reference

Public Attributes

- unsigned int buildFlags
- OptixBuildOperation operation
- OptixMotionOptions motionOptions

4.3.1 Detailed Description

Build options for acceleration structures.

See Also

optixAccelComputeMemoryUsage(), optixAccelBuild()

4.3.2 Member Data Documentation

4.3.2.1 unsigned int OptixAccelBuildOptions::buildFlags

Combinations of OptixBuildFlags.

4.3.2.2 OptixMotionOptions OptixAccelBuildOptions::motionOptions

Options for motion.

4.3.2.3 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.

4.4 OptixAccelEmitDesc Struct Reference

Public Attributes

- CUdeviceptr result
- OptixAccelPropertyType type

4.4.1 Detailed Description

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

See Also

optixAccelBuild()

4.4.2 Member Data Documentation

4.4.2.1 CUdeviceptr OptixAccelEmitDesc::result

Output buffer for the properties.

4.4.2.2 OptixAccelPropertyType OptixAccelEmitDesc::type

Requested property.

4.5 OptixAccelRelocationInfo Struct Reference

Public Attributes

- unsigned long long info [4]

4.5.1 Detailed Description

Used to store information realated to relocation of acceleration structures.

See Also

optixAccelGetRelocationInfo(), optixAccelCheckRelocationCompatibility(), optixAccelRelocate()

4.5.2 Member Data Documentation

4.5.2.1 unsigned long long OptixAccelRelocationInfo::info[4]

Opaque data, used internally, should not be modified.

4.6 OptixBuildInput Struct Reference

Public Attributes

- OptixBuildInputType type
- OptixBuildInputTriangleArray triangleArray
- OptixBuildInputCustomPrimitiveArray aabbArray
- OptixBuildInputInstanceArray instanceArray

4.6.1 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()

4.6.2 Member Data Documentation

4.6.2.1 OptixBuildInputCustomPrimitiveArray OptixBuildInput::aabbArray

Custome primitive inputs.

4.6.2.2 OptixBuildInputInstanceArray OptixBuildInput::instanceArray

Instance and instance pointer inputs.

4.6.2.3 OptixBuildInputTriangleArray OptixBuildInput::triangleArray

Triangle inputs.

4.6.2.4 OptixBuildInputType OptixBuildInput::type

The type of the build input.

4.7 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

4.7.1 Detailed Description

Custom primitive inputs

See Also

OptixBuildInput::aabbArray

4.7.2 Member Data Documentation

4.7.2.1 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.

4.7.2.2 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

4.7.2.3 unsigned int OptixBuildInputCustomPrimitiveArray::numPrimitives

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

4.7.2.4 unsigned int OptixBuildInputCustomPrimitiveArray::numSbtRecords

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

4.7.2.5 unsigned int OptixBuildInputCustomPrimitiveArray::primitiveIndexOffset

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

4.7.2.6 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.

4.7.2.7 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).

4.7.2.8 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).

4.7.2.9 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).

4.8 OptixBuildInputInstanceArray Struct Reference

Public Attributes

- CUdeviceptr instances
- unsigned int numInstances
- CUdeviceptr aabbs
- unsigned int numAabbs

4.8.1 Detailed Description

Instance and instance pointer inputs

See Also

OptixBuildInput::instanceArray

4.8.2 Member Data Documentation

4.8.2.1 CUdeviceptr OptixBuildInputInstanceArray::aabbs

Optional AABBs. In OptixAabb format.

Required for traversables (OptixMatrixMotionTransform, OptixSRTMotionTransform, OptixStaticTransform) and certain configurations of motion AS as instance. Will be ignored for non-motion AS, since no AABBs are required. May be NULL in that case.

The following table illustrates this (IAS is Instance Acceleration Structure) instance type | traversable | motion AS | static AS building a motion IAS | required | required | ignored building a static IAS | required | ignored | ignored

If OptixBuildInput::type is OPTIX_BUILD_INPUT_TYPE_INSTANCE_POINTERS the unused pointers for unused aabbs may be set to NULL.

If OptixBuildInput::type is OPTIX_BUILD_INPUT_TYPE_INSTANCES this pointer must be a multiple of OPTIX_AABB_BUFFER_BYTE_ALIGNMENT. If OptixBuildInput::type is OPTIX_BUILD_INPUT_TYPE_INSTANCE_POINTERS the array elements must be a multiple of OPTIX_AABB_BUFFER_BYTE_ALIGNMENT.

Motion:

In case of motion (OptixMotionOptions::numKeys>=2), OptixMotionOptions::numKeys aabbs are expected per instance, e.g., for N instances and M motion keys: aabb[inst0][t0], aabb[inst0][t1], ..., aabb[inst0][tM-1], ..., aabb[instN-1][t0], aabb[instN-1][t1],..., aabb[instN-1][tM-1].

If OptixBuildInput::type is OPTIX_BUILD_INPUT_TYPE_INSTANCES aabbs must be a device pointer to an array of N $*$ M $*$ 6 floats.

If OptixBuildInput::type is OPTIX_BUILD_INPUT_TYPE_INSTANCE_POINTERS aabbs must be a device pointer to an array of N device pointers, each pointing to an array of M $*$ 6 floats in OptixAabb format. Pointers may be NULL if the aabbs are not required. Hence, if the second instance (inst1) points to a static GAS, aabbs are not required for that instance. While being ignored, aabbs must still be a device pointer to an array of N elements.

In case of OPTIX_BUILD_INPUT_TYPE_INSTANCES, the second element (with a size of M $*$ 6 floats) will be ignored. In case of OPTIX_BUILD_INPUT_TYPE_INSTANCE_POINTERS, the second element (with a size of pointer to M $*$ 6 floats) can be NULL.

4.8.2.2 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.

4.8.2.3 unsigned int OptixBuildInputInstanceArray::numAabbs

number of aabbs, in case of motion, this needs to match numInstances multiplied with OptixMotionOptions::numKeys

4.8.2.4 unsigned int OptixBuildInputInstanceArray::numInstances

Number of elements in OptixBuildInputInstanceArray::instances.

4.9 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

4.9.1 Detailed Description

Triangle inputs

See Also

OptixBuildInput::triangleArray

4.9.2 Member Data Documentation

4.9.2.1 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

4.9.2.2 CUdeviceptr OptixBuildInputTriangleArray::indexBuffer

Optional pointer to array of 16 or 32-bit int triplets, one triplet per triangle.

4.9.2.3 OptixIndicesFormat OptixBuildInputTriangleArray::indexFormat

See Also

OptixIndicesFormat

4.9.2.4 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.

4.9.2.5 unsigned int OptixBuildInputTriangleArray::numIndexTriplets

Size of array in OptixBuildInputTriangleArray::indexBuffer. Needs to be zero if indexBuffer is `nullptr`.

4.9.2.6 unsigned int OptixBuildInputTriangleArray::numSbtRecords

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

4.9.2.7 unsigned int OptixBuildInputTriangleArray::numVertices

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

4.9.2.8 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

4.9.2.9 unsigned int OptixBuildInputTriangleArray::primitiveIndexOffset

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

4.9.2.10 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.

4.9.2.11 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).

4.9.2.12 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).

4.9.2.13 const CUdeviceptr* OptixBuildInputTriangleArray::vertexBuffers

Points 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 triangles).

4.9.2.14 OptixVertexFormat OptixBuildInputTriangleArray::vertexFormat

See Also

OptixVertexFormat

4.9.2.15 unsigned int OptixBuildInputTriangleArray::vertexStrideInBytes

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

4.10 OptixDenoiserOptions Struct Reference

Public Attributes

- OptixDenoiserInputKind inputKind

4.10.1 Detailed Description

Options used by the denoiser

See Also

optixDenoiserCreate()

4.10.2 Member Data Documentation

4.10.2.1 OptixDenoiserInputKind OptixDenoiserOptions::inputKind

The kind of denoiser input.

4.11 OptixDenoiserParams Struct Reference

4.11.1 Detailed Description

Various parameters used by the denoiser

See Also

optixDenoiserInvoke()

4.12 OptixDenoiserSizes Struct Reference

4.12.1 Detailed Description

Various sizes related to the denoiser.

See Also

optixDenoiserComputeMemoryResources()

4.13 OptixDeviceContextOptions Struct Reference

Public Attributes

- OptixLogCallback logCallbackFunction
- void * logCallbackData
- int logCallbackLevel

4.13.1 Detailed Description

Parameters used for optixDeviceContextCreate()

See Also

optixDeviceContextCreate()

4.13.2 Member Data Documentation

4.13.2.1 void* OptixDeviceContextOptions::logCallbackData

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

4.13.2.2 OptixLogCallback OptixDeviceContextOptions::logCallbackFunction

Function pointer used when OptiX wishes to generate messages.

4.13.2.3 int OptixDeviceContextOptions::logCallbackLevel

Maximum callback level to generate message for (see OptixLogCallback)

4.14 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(∗ optixModuleDestroy)(OptixModule module)

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)

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, const OptixDenoiserOptions ∗options, OptixDenoiser ∗returnHandle)
- OptixResult(∗ optixDenoiserDestroy)(OptixDenoiser handle)
- OptixResult(∗ optixDenoiserComputeMemoryResources)(const OptixDenoiser handle, unsigned int maximumOutputWidth, unsigned int maximumOutputHeight, OptixDenoiserSizes ∗returnSizes)
- OptixResult(∗ optixDenoiserSetup)(OptixDenoiser denoiser, CUstream stream, unsigned int outputWidth, unsigned int outputHeight, 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 OptixImage2D ∗inputLayers, unsigned int numInputLayers, unsigned int inputOffsetX, unsigned int inputOffsetY, const OptixImage2D ∗outputLayer, CUdeviceptr scratch, size_t scratchSizeInBytes)
- OptixResult(∗ optixDenoiserSetModel)(OptixDenoiser handle, OptixDenoiserModelKind kind, void ∗data, size_t sizeInBytes)
- OptixResult(∗ optixDenoiserComputeIntensity)(OptixDenoiser handle, CUstream stream, const OptixImage2D ∗inputImage, CUdeviceptr outputIntensity, CUdeviceptr scratch, size_t scratchSizeInBytes)

4.14.1 Detailed Description

The function table containing all API functions.

See optixInit() and optixInitWithHandle().

4.14.2 Member Data Documentation

4.14.2.1 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().

4.14.2.2 OptixResult(* OptixFunctionTable::optixAccelCheckRelocationCompatibility)(OptixDeviceContext context, const OptixAccelRelocationInfo *info, int *compatible)

See optixAccelCheckRelocationCompatibility().

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

See optixAccelCompact().

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

See optixAccelComputeMemoryUsage().

4.14.2.5 OptixResult(* OptixFunctionTable::optixAccelGetRelocationInfo)(OptixDeviceContext context, OptixTraversableHandle handle, OptixAccelRelocationInfo *info)

See optixAccelGetRelocationInfo().

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

See optixAccelRelocate().

4.14.2.7 OptixResult(* OptixFunctionTable::optixConvertPointerToTraversableHandle)(OptixDeviceContext onDevice, CUdeviceptr pointer, OptixTraversableType traversableType, OptixTraversableHandle *traversableHandle)

See optixConvertPointerToTraversableHandle().

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

See optixDenoiserComputeIntensity().

4.14.2.9 OptixResult(∗ OptixFunctionTable::optixDenoiserComputeMemoryResources)(const OptixDenoiser handle, unsigned int maximumOutputWidth, unsigned int maximumOutputHeight, OptixDenoiserSizes ∗returnSizes)

See optixDenoiserComputeMemoryResources().

4.14.2.10 OptixResult(∗ OptixFunctionTable::optixDenoiserCreate)(OptixDeviceContext context, const OptixDenoiserOptions ∗options, OptixDenoiser ∗returnHandle)

See optixDenoiserCreate().

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

See optixDenoiserDestroy().

4.14.2.12 OptixResult(∗ OptixFunctionTable::optixDenoiserInvoke)(OptixDenoiser denoiser, CUstream stream, const OptixDenoiserParams ∗params, CUdeviceptr denoiserState, size_t denoiserStateSizeInBytes, const OptixImage2D ∗inputLayers, unsigned int numInputLayers, unsigned int inputOffsetX, unsigned int inputOffsetY, const OptixImage2D ∗outputLayer, CUdeviceptr scratch, size_t scratchSizeInBytes)

See optixDenoiserInvoke().

4.14.2.13 OptixResult(∗ OptixFunctionTable::optixDenoiserSetModel)(OptixDenoiser handle, OptixDenoiserModelKind kind, void ∗data, size_t sizeInBytes)

See optixDenoiserSetModel().

4.14.2.14 OptixResult(∗ OptixFunctionTable::optixDenoiserSetup)(OptixDenoiser denoiser, CUstream stream, unsigned int outputWidth, unsigned int outputHeight, CUdeviceptr state, size_t stateSizeInBytes, CUdeviceptr scratch, size_t scratchSizeInBytes)

See optixDenoiserSetup().

4.14.2.15 OptixResult(∗ OptixFunctionTable::optixDeviceContextCreate)(CUcontext fromContext, const OptixDeviceContextOptions ∗options, OptixDeviceContext ∗context)

See optixDeviceContextCreate().

4.14.2.16 OptixResult(∗ OptixFunctionTable::optixDeviceContextDestroy)(OptixDeviceContext context)

See optixDeviceContextDestroy().

4.14.2.17 OptixResult(∗ OptixFunctionTable::optixDeviceContextGetCacheDatabaseSizes)(OptixDeviceContext context, size_t ∗lowWaterMark, size_t ∗highWaterMark)

See optixDeviceContextGetCacheDatabaseSizes().

4.14.2.18 OptixResult(∗ OptixFunctionTable::optixDeviceContextGetCacheEnabled)(OptixDeviceContext context, int ∗enabled)

See optixDeviceContextGetCacheEnabled().

4.14.2.19 OptixResult(∗ OptixFunctionTable::optixDeviceContextGetCacheLocation)(OptixDeviceContext context, char ∗location, size_t locationSize)

See optixDeviceContextGetCacheLocation().

4.14.2.20 OptixResult(∗ OptixFunctionTable::optixDeviceContextGetProperty)(OptixDeviceContext context, OptixDeviceProperty property, void ∗value, size_t sizeInBytes)

See optixDeviceContextGetProperty().

4.14.2.21 OptixResult(∗ OptixFunctionTable::optixDeviceContextSetCacheDatabaseSizes)(OptixDeviceContext context, size_t lowWaterMark, size_t highWaterMark)

See optixDeviceContextSetCacheDatabaseSizes().

4.14.2.22 OptixResult(∗ OptixFunctionTable::optixDeviceContextSetCacheEnabled)(OptixDeviceContext context, int enabled)

See optixDeviceContextSetCacheEnabled().

4.14.2.23 OptixResult(∗ OptixFunctionTable::optixDeviceContextSetCacheLocation)(OptixDeviceContext context, const char ∗location)

See optixDeviceContextSetCacheLocation().

4.14.2.24 OptixResult(∗ OptixFunctionTable::optixDeviceContextSetLogCallback)(OptixDeviceContext context, OptixLogCallback callbackFunction, void ∗callbackData, unsigned int callbackLevel)

See optixDeviceContextSetLogCallback().

4.14.2.25 const char*(* OptixFunctionTable::optixGetErrorName)(OptixResult result)

See optixGetErrorName().

4.14.2.26 const char*(* OptixFunctionTable::optixGetErrorString)(OptixResult result)

See optixGetErrorString().

4.14.2.27 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().

4.14.2.28 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().

4.14.2.29 OptixResult(* OptixFunctionTable::optixModuleDestroy)(OptixModule module)

See optixModuleDestroy().

4.14.2.30 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().

4.14.2.31 OptixResult(* OptixFunctionTable::optixPipelineDestroy)(OptixPipeline pipeline)

See optixPipelineDestroy().

4.14.2.32 OptixResult(* OptixFunctionTable::optixPipelineSetStackSize)(OptixPipeline pipeline, unsigned int directCallableStackSizeFromTraversal, unsigned int directCallableStackSizeFromState, unsigned int continuationStackSize, unsigned int maxTraversableGraphDepth)

See optixPipelineSetStackSize().

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

See optixProgramGroupCreate().

4.14.2.34 OptixResult(∗ OptixFunctionTable::optixProgramGroupDestroy)(OptixProgramGroup programGroup)

See optixProgramGroupDestroy().

4.14.2.35 OptixResult(∗ OptixFunctionTable::optixProgramGroupGetStackSize)(OptixProgramGroup programGroup, OptixStackSizes ∗stackSizes)

See optixProgramGroupGetStackSize().

4.14.2.36 OptixResult(∗ OptixFunctionTable::optixSbtRecordPackHeader)(OptixProgramGroup programGroup, void ∗sbtRecordHeaderHostPointer)

See optixConvertPointerToTraversableHandle().

4.15 OptixImage2D Struct Reference

Public Attributes

- CUdeviceptr data
- unsigned int width
- unsigned int height
- unsigned int rowStrideInBytes
- unsigned int pixelStrideInBytes
- OptixPixelFormat format

4.15.1 Detailed Description

Image descriptor used by the denoiser.

See Also

optixDenoiserInvoke(), optixDenoiserComputeIntensity()

4.15.2 Member Data Documentation

4.15.2.1 CUdeviceptr OptixImage2D::data

Pointer to the actual pixel data.

4.15.2.2 OptixPixelFormat OptixImage2D::format

Pixel format.

4.15.2.3 unsigned int OptixImage2D::height

Height of the image (in pixels)

4.15.2.4 unsigned int OptixImage2D::pixelStrideInBytes

Stride between subsequent pixels of the image (in bytes). For now, only 0 or the value that corresponds to a dense packing of pixels (no gaps) is supported.

4.15.2.5 unsigned int OptixImage2D::rowStrideInBytes

Stride between subsequent rows of the image (in bytes).

4.15.2.6 unsigned int OptixImage2D::width

Width of the image (in pixels)

4.16 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]

4.16.1 Detailed Description

Instances

See Also

OptixBuildInputInstanceArray::instances

4.16.2 Member Data Documentation

4.16.2.1 unsigned int OptixInstance::flags

Any combination of OptixInstanceFlags is allowed.

4.16.2.2 unsigned int OptixInstance::instanceId

Application supplied ID. The maximal ID can be queried using OPTIX_DEVICE_PROPERTY_LIMIT_MAX_INSTANCE_ID.

4.16.2.3 unsigned int OptixInstance::pad[2]

round up to 80-byte, to ensure 16-byte alignment

4.16.2.4 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.

4.16.2.5 float OptixInstance::transform[12]

affine world-to-object transformation as 3x4 matrix in row-major layout

4.16.2.6 OptixTraversableHandle OptixInstance::traversableHandle

Set with an OptixTraversableHandle.

4.16.2.7 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.

4.17 OptixMatrixMotionTransform Struct Reference

Public Attributes

- OptixTraversableHandle child
- OptixMotionOptions motionOptions
- unsigned int pad [3]
- float transform [2][12]

4.17.1 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()

4.17.2 Member Data Documentation

4.17.2.1 OptixTraversableHandle OptixMatrixMotionTransform::child

The traversable that is transformed by this transformation.

4.17.2.2 OptixMotionOptions OptixMatrixMotionTransform::motionOptions

The motion options for this transformation.

4.17.2.3 unsigned int OptixMatrixMotionTransform::pad[3]

Padding to make the transformation 16 byte aligned.

4.17.2.4 float OptixMatrixMotionTransform::transform[2][12]

Affine object-to-world transformation as 3x4 matrix in row-major layout.

4.18 OptixModuleCompileOptions Struct Reference

Public Attributes

- int maxRegisterCount
- OptixCompileOptimizationLevel optLevel
- OptixCompileDebugLevel debugLevel

4.18.1 Detailed Description

Compilation options for module

See Also

optixModuleCreateFromPTX()

4.18.2 Member Data Documentation

4.18.2.1 OptixCompileDebugLevel OptixModuleCompileOptions::debugLevel

Generate debug information.

4.18.2.2 int OptixModuleCompileOptions::maxRegisterCount

Maximum number of registers allowed when compiling to SASS. Set to 0 for no explicit limit. May vary within a pipeline.

4.18.2.3 OptixCompileOptimizationLevel OptixModuleCompileOptions::optLevel

Optimization level. May vary within a pipeline.

4.19 OptixMotionOptions Struct Reference

Public Attributes

- unsigned short numKeys
- unsigned short flags
- float timeBegin
- float timeEnd

4.19.1 Detailed Description

Motion options

See Also

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

4.19.2 Member Data Documentation

4.19.2.1 unsigned short OptixMotionOptions::flags

Combinations of OptixMotionFlags.

4.19.2.2 unsigned short OptixMotionOptions::numKeys

If numKeys $>$ 1, motion is enabled. timeBegin, timeEnd and flags are all ignored when motion is disabled.

4.19.2.3 float OptixMotionOptions::timeBegin

Point in time where motion starts.

4.19.2.4 float OptixMotionOptions::timeEnd

Point in time where motion ends.

4.20 OptixPipelineCompileOptions Struct Reference

Public Attributes

- int usesMotionBlur
- unsigned int traversableGraphFlags
- int numPayloadValues
- int numAttributeValues
- unsigned int exceptionFlags
- const char $*$ pipelineLaunchParamsVariableName

4.20.1 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()

4.20.2 Member Data Documentation

4.20.2.1 unsigned int OptixPipelineCompileOptions::exceptionFlags

A bitmask of OptixExceptionFlags indicating which exceptions are enabled.

4.20.2.2 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]

4.20.2.3 int OptixPipelineCompileOptions::numPayloadValues

How much storage, in 32b words, to make available for the payload, [0..8].

4.20.2.4 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.

4.20.2.5 unsigned int OptixPipelineCompileOptions::traversableGraphFlags

Traversable graph bitfield. See OptixTraversableGraphFlags.

4.20.2.6 int OptixPipelineCompileOptions::usesMotionBlur

Boolean value indicating whether motion blur could be used.

4.21 OptixPipelineLinkOptions Struct Reference

Public Attributes

- unsigned int maxTraceDepth
- OptixCompileDebugLevel debugLevel
- int overrideUsesMotionBlur

4.21.1 Detailed Description

Link options for a pipeline

See Also

optixPipelineCreate()

4.21.2 Member Data Documentation

4.21.2.1 OptixCompileDebugLevel OptixPipelineLinkOptions::debugLevel

Generate debug information.

4.21.2.2 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.

4.21.2.3 int OptixPipelineLinkOptions::overrideUsesMotionBlur

Boolean value that customizes the pipeline to enable or disable motion blur. If enabled all modules must have specified the usesMotionBlur flag in OptixPipelineCompileOptions.

4.22 OptixProgramGroupCallables Struct Reference

Public Attributes

- OptixModule moduleDC
- const char * entryFunctionNameDC
- OptixModule moduleCC
- const char * entryFunctionNameCC

4.22.1 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

4.22.2 Member Data Documentation

4.22.2.1 const char* OptixProgramGroupCallables::entryFunctionNameCC

Entry function name of the continuation callable (CC) program.

4.22.2.2 const char* OptixProgramGroupCallables::entryFunctionNameDC

Entry function name of the direct callable (DC) program.

4.22.2.3 OptixModule OptixProgramGroupCallables::moduleCC

Module holding the continuation callable (CC) program.

4.22.2.4 OptixModule OptixProgramGroupCallables::moduleDC

Module holding the direct callable (DC) program.

4.23 OptixProgramGroupDesc Struct Reference

Public Attributes

- OptixProgramGroupKind kind
- unsigned int flags
- OptixProgramGroupSingleModule raygen
- OptixProgramGroupSingleModule miss
- OptixProgramGroupSingleModule exception
- OptixProgramGroupCallables callables
- OptixProgramGroupHitgroup hitgroup

4.23.1 Detailed Description

Descriptor for program groups.

4.23.2 Member Data Documentation

4.23.2.1 OptixProgramGroupCallables OptixProgramGroupDesc::callables

See Also

OPTIX_PROGRAM_GROUP_KIND_CALLABLES

4.23.2.2 OptixProgramGroupSingleModule OptixProgramGroupDesc::exception

See Also

OPTIX_PROGRAM_GROUP_KIND_EXCEPTION

4.23.2.3 unsigned int OptixProgramGroupDesc::flags

See OptixProgramGroupFlags.

4.23.2.4 OptixProgramGroupHitgroup OptixProgramGroupDesc::hitgroup

See Also

OPTIX_PROGRAM_GROUP_KIND_HITGROUP

4.23.2.5 OptixProgramGroupKind OptixProgramGroupDesc::kind

The kind of program group.

4.23.2.6 OptixProgramGroupSingleModule OptixProgramGroupDesc::miss

See Also

OPTIX_PROGRAM_GROUP_KIND_MISS

4.23.2.7 OptixProgramGroupSingleModule OptixProgramGroupDesc::raygen

See Also

OPTIX_PROGRAM_GROUP_KIND_RAYGEN

4.24 OptixProgramGroupHitgroup Struct Reference

Public Attributes

- OptixModule moduleCH
- const char * entryFunctionNameCH

- OptixModule moduleAH
- const char * entryFunctionNameAH
- OptixModule moduleIS
- const char * entryFunctionNameIS

4.24.1 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

4.24.2 Member Data Documentation

4.24.2.1 const char* OptixProgramGroupHitgroup::entryFunctionNameAH

Entry function name of the any hit (AH) program.

4.24.2.2 const char* OptixProgramGroupHitgroup::entryFunctionNameCH

Entry function name of the closest hit (CH) program.

4.24.2.3 const char* OptixProgramGroupHitgroup::entryFunctionNameIS

Entry function name of the intersection (IS) program.

4.24.2.4 OptixModule OptixProgramGroupHitgroup::moduleAH

Module holding the any hit (AH) program.

4.24.2.5 OptixModule OptixProgramGroupHitgroup::moduleCH

Module holding the closest hit (CH) program.

4.24.2.6 OptixModule OptixProgramGroupHitgroup::moduleIS

Module holding the intersection (Is) program.

4.25 OptixProgramGroupOptions Struct Reference

Public Attributes

- int placeholder

4.25.1 Detailed Description

Program group options

See Also

optixProgramGroupCreate()

4.25.2 Member Data Documentation

4.25.2.1 int OptixProgramGroupOptions::placeholder

Currently no options, so include a placeholder.

4.26 OptixProgramGroupSingleModule Struct Reference

Public Attributes

- OptixModule module
- const char * entryFunctionName

4.26.1 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

4.26.2 Member Data Documentation

4.26.2.1 const char* OptixProgramGroupSingleModule::entryFunctionName

Entry function name of the single program.

4.26.2.2 OptixModule OptixProgramGroupSingleModule::module

Module holding single program.

4.27 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

4.27.1 Detailed Description

Describes the shader binding table (SBT)

See Also

optixLaunch()

4.27.2 Member Data Documentation

4.27.2.1 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.

4.27.2.2 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.

4.27.2.3 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.

4.27.2.4 CUdeviceptr OptixShaderBindingTable::exceptionRecord

Device address of the SBT record of the exception program. The address must be a multiple of OPTIX_SBT_RECORD_ALIGNMENT.

4.27.2.5 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.

4.27.2.6 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.

4.27.2.7 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.

4.27.2.8 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.

4.27.2.9 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.

4.27.2.10 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.

4.27.2.11 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.

4.28 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**

4.28.1 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.

```
[  sx   a   b  pvx ]
```

The scaling matrix S = [0 sy c pvy] defines an affine transformation that can include scale, shear, and a [0 0 sz pvz]

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.

```
[  1  0  0 tx ]
```

The translation T = [0 1 0 ty] defines another translation that is applied after the rotation. Typically, this [0 0 1 tz]

translation includes the inverse translation from the matrix S to reverse its effect.

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()

4.29 OptixSRTMotionTransform Struct Reference

Public Attributes

- OptixTraversableHandle child
- OptixMotionOptions motionOptions
- unsigned int pad [3]
- OptixSRTData srtData [2]

4.29.1 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()

4.29.2 Member Data Documentation

4.29.2.1 OptixTraversableHandle OptixSRTMotionTransform::child

The traversable transformed by this transformation.

4.29.2.2 OptixMotionOptions OptixSRTMotionTransform::motionOptions

The motion options for this transformation.

4.29.2.3 unsigned int OptixSRTMotionTransform::pad[3]

Padding to make the SRT data 16 byte aligned.

4.29.2.4 OptixSRTData OptixSRTMotionTransform::srtData[2]

The actual SRT data describing the transformation.

4.30 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

4.30.1 Detailed Description

Describes the stack size requirements of a program group.

See Also

optixProgramGroupGetStackSize()

4.30.2 Member Data Documentation

4.30.2.1 unsigned int OptixStackSizes::cssAH

Continuation stack size of AH programs in bytes.

4.30.2.2 unsigned int OptixStackSizes::cssCC

Continuation stack size of CC programs in bytes.

4.30.2.3 unsigned int OptixStackSizes::cssCH

Continuation stack size of CH programs in bytes.

4.30.2.4 unsigned int OptixStackSizes::cssIS

Continuation stack size of IS programs in bytes.

4.30.2.5 unsigned int OptixStackSizes::cssMS

Continuation stack size of MS programs in bytes.

4.30.2.6 unsigned int OptixStackSizes::cssRG

Continuation stack size of RG programs in bytes.

4.30.2.7 unsigned int OptixStackSizes::dssDC

Direct stack size of DC programs in bytes.

4.31 OptixStaticTransform Struct Reference

Public Attributes

- OptixTraversableHandle child
- unsigned int pad [2]
- float transform [12]
- float invTransform [12]

4.31.1 Detailed Description

Static transform

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

See Also

optixConvertPointerToTraversableHandle()

4.31.2 Member Data Documentation

4.31.2.1 OptixTraversableHandle OptixStaticTransform::child

The traversable transformed by this transformation.

4.31.2.2 float OptixStaticTransform::invTransform[12]

Affine object-to-world transformation as 3x4 matrix in row-major layout.

4.31.2.3 unsigned int OptixStaticTransform::pad[2]

Padding to make the transformations 16 byte aligned.

4.31.2.4 float OptixStaticTransform::transform[12]

Affine world-to-object transformation as 3x4 matrix in row-major layout.

5 File Documentation

5.1 optix.h File Reference

Macros

- #define OPTIX_VERSION 70000

5.1.1 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

5.1.2 Macro Definition Documentation

5.1.2.1 #define OPTIX_VERSION 70000

The OptiX version.

- major = OPTIX_VERSION/10000
- minor = (OPTIX_VERSION%10000)/100
- micro = OPTIX_VERSION%100

5.2 optix_7_device.h File Reference

Functions

- 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)
- 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, unsigned int &p0)
- 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, unsigned int &p0, unsigned int &p1)
- 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, unsigned int &p0, unsigned int &p1, unsigned int &p2)
- 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, unsigned int &p0, unsigned int &p1, unsigned int &p2, unsigned int &p3)

- 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, unsigned int &p0, unsigned int &p1, unsigned int &p2, unsigned int &p3, unsigned int &p4)
- 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, unsigned int &p0, unsigned int &p1, unsigned int &p2, unsigned int &p3, unsigned int &p4, unsigned int &p5)
- 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, unsigned int &p0, unsigned int &p1, unsigned int &p2, unsigned int &p3, unsigned int &p4, unsigned int &p5, unsigned int &p6)
- 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, unsigned int &p0, unsigned int &p1, unsigned int &p2, unsigned int &p3, unsigned int &p4, unsigned int &p5, unsigned int &p6, unsigned int &p7)
- 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__ 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 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__ void optixGetTriangleVertexData (OptixTraversableHandle gas, unsigned int primIdx, unsigned int sbtGASIndex, float time, float3 data[3])
- 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__ 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 optixGetInstanceId ()
- static __forceinline__ __device__ unsigned int optixGetInstanceIndex ()
- static __forceinline__ __device__ unsigned int optixGetHitKind ()
- static __forceinline__ __device__ bool optixIsTriangleHit ()
- static __forceinline__ __device__ bool optixIsTriangleFrontFaceHit ()
- static __forceinline__ __device__ bool optixIsTriangleBackFaceHit ()

- 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 ()

5.2.1 Detailed Description

OptiX public API header.

Author

NVIDIA Corporation OptiX public API Reference - Device API declarations

5.3 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 optixModuleDestroy (OptixModule module)
- 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, const OptixDenoiserOptions *options, OptixDenoiser *denoiser)
- OptixResult optixDenoiserSetModel (OptixDenoiser denoiser, OptixDenoiserModelKind kind, void *data, size_t sizeInBytes)
- 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 outputWidth, unsigned int outputHeight, 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 OptixImage2D *inputLayers, unsigned int numInputLayers, unsigned int inputOffsetX, unsigned int inputOffsetY, const OptixImage2D *outputLayer, CUdeviceptr scratch, size_t scratchSizeInBytes)
- OptixResult optixDenoiserComputeIntensity (OptixDenoiser denoiser, CUstream stream, const OptixImage2D *inputImage, CUdeviceptr outputIntensity, CUdeviceptr scratch, size_t scratchSizeInBytes)

5.3.1 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

5.4 optix_7_types.h File Reference

Classes

- struct OptixDeviceContextOptions
- struct OptixBuildInputTriangleArray
- 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 OptixDenoiserParams
- struct OptixDenoiserSizes
- struct OptixModuleCompileOptions
- struct OptixProgramGroupSingleModule
- struct OptixProgramGroupHitgroup
- struct OptixProgramGroupCallables
- struct OptixProgramGroupDesc
- struct OptixProgramGroupOptions
- struct OptixPipelineCompileOptions
- struct OptixPipelineLinkOptions
- struct OptixShaderBindingTable
- struct OptixStackSizes

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

Typedefs

- typedef unsigned int 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 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 struct OptixDeviceContextOptions OptixDeviceContextOptions

- typedef enum OptixGeometryFlags OptixGeometryFlags
- typedef enum OptixHitKind OptixHitKind
- typedef enum OptixIndicesFormat OptixIndicesFormat
- typedef enum OptixVertexFormat OptixVertexFormat
- typedef struct OptixBuildInputTriangleArray OptixBuildInputTriangleArray
- 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 OptixDenoiserInputKind OptixDenoiserInputKind
- typedef enum OptixDenoiserModelKind OptixDenoiserModelKind
- typedef struct OptixDenoiserOptions OptixDenoiserOptions
- 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 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 ABI_ID, unsigned int numOptions, OptixQueryFunctionTableOptions *, const void **, void *functionTable, size_t sizeOfTable)

Enumerations

- enum OptixResult
- 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 OptixGeometryFlags {
  OPTIX_GEOMETRY_FLAG_NONE = 0,
  OPTIX_GEOMETRY_FLAG_DISABLE_ANYHIT = 1u << 0,
  OPTIX_GEOMETRY_FLAG_REQUIRE_SINGLE_ANYHIT_CALL = 1u << 1 }

- enum OptixHitKind {
  OPTIX_HIT_KIND_TRIANGLE_FRONT_FACE = 0xFE,
  OPTIX_HIT_KIND_TRIANGLE_BACK_FACE = 0xFF }
- enum OptixIndicesFormat {
  OPTIX_INDICES_FORMAT_UNSIGNED_SHORT3 = 0x2102,
  OPTIX_INDICES_FORMAT_UNSIGNED_INT3 = 0x2103 }
- enum OptixVertexFormat {
  OPTIX_VERTEX_FORMAT_FLOAT3 = 0x2121,
  OPTIX_VERTEX_FORMAT_FLOAT2 = 0x2122,
  OPTIX_VERTEX_FORMAT_HALF3 = 0x2123,
  OPTIX_VERTEX_FORMAT_HALF2 = 0x2124 }
- 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 }
- 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,
  OPTIX_INSTANCE_FLAG_DISABLE_TRANSFORM = 1u << 6 }
- enum OptixBuildFlags {
  OPTIX_BUILD_FLAG_NONE = 0,
  OPTIX_BUILD_FLAG_ALLOW_UPDATE = 1u << 0 ,
  OPTIX_BUILD_FLAG_ALLOW_RANDOM_VERTEX_ACCESS = 1u << 4 }
- enum OptixBuildOperation {
  OPTIX_BUILD_OPERATION_BUILD = 0x2161,
  OPTIX_BUILD_OPERATION_UPDATE = 0x2162 }
- enum OptixMotionFlags
- 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_HALF3 = 0x2201,
  OPTIX_PIXEL_FORMAT_HALF4 = 0x2202,
  OPTIX_PIXEL_FORMAT_FLOAT3 = 0x2203,
  OPTIX_PIXEL_FORMAT_FLOAT4 = 0x2204,
  OPTIX_PIXEL_FORMAT_UCHAR3 = 0x2205,
  OPTIX_PIXEL_FORMAT_UCHAR4 = 0x2206 }
- enum OptixDenoiserInputKind
- enum OptixDenoiserModelKind {
  OPTIX_DENOISER_MODEL_KIND_USER = 0x2321,

OPTIX_DENOISER_MODEL_KIND_LDR = 0x2322,
OPTIX_DENOISER_MODEL_KIND_HDR = 0x2323 }

- 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_LEVEL_0 = 0,
  OPTIX_COMPILE_OPTIMIZATION_LEVEL_1 = 1,
  OPTIX_COMPILE_OPTIMIZATION_LEVEL_2 = 2,
  OPTIX_COMPILE_OPTIMIZATION_LEVEL_3 = 3 }
- enum OptixCompileDebugLevel {
  OPTIX_COMPILE_DEBUG_LEVEL_NONE = 0,
  OPTIX_COMPILE_DEBUG_LEVEL_LINEINFO = 1,
  OPTIX_COMPILE_DEBUG_LEVEL_FULL = 2 }
- 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 }
- 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 }

5.4.1 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

5.5 optix_function_table.h File Reference

Classes

- struct OptixFunctionTable

Macros

- #define OPTIX_ABI_VERSION 22

Typedefs

- typedef struct OptixFunctionTable OptixFunctionTable

5.5.1 Detailed Description

OptiX public API header.

Author

NVIDIA Corporation

5.5.2 Macro Definition Documentation

5.5.2.1 #define OPTIX_ABI_VERSION 22

The OptiX ABI version.

5.6 optix_function_table_definition.h File Reference

Variables

- OptixFunctionTable g_optixFunctionTable

5.6.1 Detailed Description

OptiX public API header.

Author

NVIDIA Corporation

5.7 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)

5.7.1 Detailed Description

OptiX public API header.

Author

NVIDIA Corporation

5.8 optix_stubs.h File Reference

Functions

- OptixResult optixInitWithHandle (void **handlePtr)
- OptixResult optixInit (void)

Variables

- OptixFunctionTable g_optixFunctionTable

5.8.1 Detailed Description

OptiX public API header.

Author

NVIDIA Corporation