roctracer

4.1.0

Generated by Doxygen 1.8.18

Fri Sep 23 2022 21:37:40

Chapter 1

ROC Tracer API Specification

1.1 Introduction

ROCtracer library, Runtimes Generic Callback/Activity APIs.

The goal of the implementation is to provide a generic independent from specific runtime profiler to trace API and asynchronous activity.

The API provides functionality for registering the runtimes API callbacks and asynchronous activity records pool support.

Chapter 2

Todo List

Global roctracer_op_string (uint32_t domain, uint32_t op, uint32_t kind) ROCTRACER_VERSION_4_1

Define kind.

Chapter 3

Module Documentation

3.1 Symbol Versions

The names used for the shared library versioned symbols.

Macros

- #define ROCTRACER_VERSION_4_1

  The function was introduced in version 4.1 of the interface and has the symbol version string of `"ROCTRACER_4.1"`.

3.1.1 Detailed Description

The names used for the shared library versioned symbols.

Every function is annotated with one of the version macros defined in this section. Each macro specifies a corresponding symbol version string. After dynamically loading the shared library with `dlopen`, the address of each function can be obtained using `dlvsym` with the name of the function and its corresponding symbol version string. An error will be reported by `dlvsym` if the installed library does not support the version for the function specified in this version of the interface.

3.1.2 Macro Definition Documentation

3.1.2.1 ROCTRACER_VERSION_4_1

```
#define ROCTRACER_VERSION_4_1
```

The function was introduced in version 4.1 of the interface and has the symbol version string of `"ROCTRACER_4.1"`.

3.2 Versioning

Version information about the interface and the associated installed library.

Macros

- #define ROCTRACER_VERSION_MAJOR 4

  The major version of the interface as a macro so it can be used by the preprocessor.
- #define ROCTRACER_VERSION_MINOR 1

  The minor version of the interface as a macro so it can be used by the preprocessor.

Functions

- ROCTRACER_API uint32_t roctracer_version_major () ROCTRACER_VERSION_4_1

  Query the major version of the installed library.
- ROCTRACER_API uint32_t roctracer_version_minor () ROCTRACER_VERSION_4_1

  Query the minor version of the installed library.

3.2.1 Detailed Description

Version information about the interface and the associated installed library.

The semantic version of the interface following semver.org rules. A client that uses this interface is only compatible with the installed library if the major version numbers match and the interface minor version number is less than or equal to the installed library minor version number.

3.2.2 Macro Definition Documentation

3.2.2.1 ROCTRACER_VERSION_MAJOR

```
#define ROCTRACER_VERSION_MAJOR 4
```

The major version of the interface as a macro so it can be used by the preprocessor.

3.2.2.2 ROCTRACER_VERSION_MINOR

```
#define ROCTRACER_VERSION_MINOR 1
```

The minor version of the interface as a macro so it can be used by the preprocessor.

3.2.3 Function Documentation

3.2.3.1 roctracer_version_major()

ROCTRACER_API uint32_t roctracer_version_major ( )

Query the major version of the installed library.

Return the major version of the installed library. This can be used to check if it is compatible with this interface version. This function can be used even when the library is not initialized.

3.2.3.2 roctracer_version_minor()

ROCTRACER_API uint32_t roctracer_version_minor ( )

Query the minor version of the installed library.

Return the minor version of the installed library. This can be used to check if it is compatible with this interface version. This function can be used even when the library is not initialized.

3.3 Status Codes

Most operations return a status code to indicate success or error.

Enumerations

- enum roctracer_status_t {
  ROCTRACER_STATUS_SUCCESS = 0, ROCTRACER_STATUS_ERROR = -1, ROCTRACER_STATUS_ERROR_INVALID_DOM
  = -2, ROCTRACER_STATUS_ERROR_INVALID_ARGUMENT = -3,
  ROCTRACER_STATUS_ERROR_DEFAULT_POOL_UNDEFINED = -4, ROCTRACER_STATUS_ERROR_DEFAULT_POOL_ALL
  = -5, ROCTRACER_STATUS_ERROR_MEMORY_ALLOCATION = -6, ROCTRACER_STATUS_ERROR_MISMATCHED_EXTER
  = -7,
  ROCTRACER_STATUS_UNINIT = 2, ROCTRACER_STATUS_BREAK = 3, ROCTRACER_STATUS_BAD_DOMAIN
  = ROCTRACER_STATUS_ERROR_INVALID_DOMAIN_ID, ROCTRACER_STATUS_BAD_PARAMETER =
  ROCTRACER_STATUS_ERROR_INVALID_ARGUMENT,
  ROCTRACER_STATUS_HIP_API_ERR = 6, ROCTRACER_STATUS_HIP_OPS_ERR = 7, ROCTRACER_STATUS_HCC_OPS_E
  = ROCTRACER_STATUS_HIP_OPS_ERR, ROCTRACER_STATUS_HSA_ERR = 7,
  ROCTRACER_STATUS_ROCTX_ERR = 8 }

  ROC Tracer API status codes.

Functions

- ROCTRACER_API const char ∗ roctracer_error_string () ROCTRACER_VERSION_4_1

  Query the textual description of the last error for the current thread.

3.3.1 Detailed Description

Most operations return a status code to indicate success or error.

3.3.2 Enumeration Type Documentation

3.3.2.1 roctracer_status_t

```
enum roctracer_status_t
```

ROC Tracer API status codes.

Enumerator

ROCTRACER_STATUS_SUCCESS	The function has executed successfully.
ROCTRACER_STATUS_ERROR	A generic error has occurred.
ROCTRACER_STATUS_ERROR_INVALID_DOMAIN_ID	The domain ID is invalid.

Enumerator

ROCTRACER_STATUS_ERROR_INVALID_ARGUMENT	An invalid argument was given to the function.
ROCTRACER_STATUS_ERROR_DEFAULT_POOL_UNDEFI←NED	No default pool is defined.
ROCTRACER_STATUS_ERROR_DEFAULT_POOL_ALREAD←Y_DEFINED	The default pool is already defined.
ROCTRACER_STATUS_ERROR_MEMORY_ALLOCATION	Memory allocation error.
ROCTRACER_STATUS_ERROR_MISMATCHED_EXTERNA←L_CORRELATION_ID	External correlation ID pop mismatch.
ROCTRACER_STATUS_UNINIT	Deprecated error code.
ROCTRACER_STATUS_BREAK	Deprecated error code.
ROCTRACER_STATUS_BAD_DOMAIN	Deprecated error code.
ROCTRACER_STATUS_BAD_PARAMETER	Deprecated error code.
ROCTRACER_STATUS_HIP_API_ERR	Deprecated error code.
ROCTRACER_STATUS_HIP_OPS_ERR	Deprecated error code.
ROCTRACER_STATUS_HCC_OPS_ERR	Deprecated error code.
ROCTRACER_STATUS_HSA_ERR	Deprecated error code.
ROCTRACER_STATUS_ROCTX_ERR	Deprecated error code.

3.3.3 Function Documentation

3.3.3.1 roctracer_error_string()

ROCTRACER_API const char* roctracer_error_string ( )

Query the textual description of the last error for the current thread.

Returns a NUL terminated string describing the error of the last ROC Tracer API call by the calling thread that did not return success. The empty string is returned if there is no previous error. The last error is not cleared.

Returns

Return the error string. The caller owns the returned string and should use `free()` to deallocate it.

3.4 Traced Runtime Domains

The ROC Tracer API can trace multiple runtime libraries. Each library can have API operations and asynchronous operations that can be traced.

Typedefs

- typedef activity_domain_t roctracer_domain_t

    Enumeration of domains that can be traced.

Functions

- ROCTRACER_API const char ∗ roctracer_op_string (uint32_t domain, uint32_t op, uint32_t kind) ROCTRACER_VERSION_4_1

    Query textual name of an operation of a domain.

- ROCTRACER_API roctracer_status_t roctracer_op_code (uint32_t domain, const char ∗str, uint32_t ∗op, uint32_t ∗kind) ROCTRACER_VERSION_4_1

    Query the operation code given a domain and the name of an operation.

- ROCTRACER_API roctracer_status_t roctracer_set_properties (roctracer_domain_t domain, void ∗properties) ROCTRACER_VERSION_4_1

    Set the properties of a domain.

3.4.1 Detailed Description

The ROC Tracer API can trace multiple runtime libraries. Each library can have API operations and asynchronous operations that can be traced.

3.4.2 Typedef Documentation

3.4.2.1 roctracer_domain_t

```
typedef activity_domain_t roctracer_domain_t
```

Enumeration of domains that can be traced.

3.4.3 Function Documentation

3.4.3.1 roctracer_op_code()

```
ROCTRACER_API roctracer_status_t roctracer_op_code (
            uint32_t domain,
            const char * str,
            uint32_t * op,
            uint32_t * kind )
```

Query the operation code given a domain and the name of an operation.

Parameters

in	*domain*	The domain being queried.
in	*str*	The NUL terminated name of the operation name being queried.
out	*op*	The operation code.
out	*kind*	If not NULL then the operation kind code.

Return values

[ROCTRACER_STATUS_SUCCESS](#)	The function has been executed successfully. `op` and `kind` have been updated.
[ROCTRACER_STATUS_ERROR_INVALID_ARGUMENT](#)	The `op` is invalid for `domain`.
[ROCTRACER_STATUS_ERROR_INVALID_DOMAIN_ID](#)	The domain is invalid or not supported.

3.4.3.2 roctracer_op_string()

```
ROCTRACER_API const char* roctracer_op_string (
            uint32_t domain,
            uint32_t op,
            uint32_t kind )
```

Query textual name of an operation of a domain.

Parameters

in	*domain*	Domain being queried.
in	*op*	Operation within `domain`.
in	*kind*	

Todo Define kind.

Returns

Returns the NUL terminated string for the operation name, or NULL if the domain or operation are invalid. The string is owned by the ROC Tracer library.

3.4.3.3 roctracer_set_properties()

```
ROCTRACER_API roctracer_status_t roctracer_set_properties (
            roctracer_domain_t domain,
            void * properties )
```

Set the properties of a domain.

Parameters

in	*domain*	The domain.
in	*properties*	The properties. Each domain defines its own type for the properties. Some domains require the properties to be set before they can be enabled.

Return values

ROCTRACER_STATUS_SUCCESS	The function has been executed successfully.

3.5 Callback API

ROC tracer provides support for runtime API callbacks and activity records logging. The API callbacks provide the API calls arguments and are called on different phases, on enter, on exit, on kernel completion.

Typedefs

- typedef activity_rtapi_callback_t roctracer_rtapi_callback_t

    Runtime API callback type.

Functions

- ROCTRACER_API roctracer_status_t roctracer_enable_op_callback (activity_domain_t domain, uint32_t op, activity_rtapi_callback_t callback, void ∗arg) ROCTRACER_VERSION_4_1

    Enable runtime API callback for a specific operation of a domain.
- ROCTRACER_API roctracer_status_t roctracer_enable_domain_callback (activity_domain_t domain, activity_↩ rtapi_callback_t callback, void ∗arg) ROCTRACER_VERSION_4_1

    Enable runtime API callback for all operations of a domain.
- ROCTRACER_API roctracer_status_t roctracer_enable_callback (activity_rtapi_callback_t callback, void ∗arg) ROCTRACER_VERSION_4_1

    Enable runtime API callback for all operations of all domains.
- ROCTRACER_API roctracer_status_t roctracer_disable_op_callback (activity_domain_t domain, uint32_t op) ROCTRACER_VERSION_4_1

    Disable runtime API callback for a specific operation of a domain.
- ROCTRACER_API roctracer_status_t roctracer_disable_domain_callback (activity_domain_t domain) ROCTRACER_VERSION_4_

    Disable runtime API callback for all operations of a domain.
- ROCTRACER_API roctracer_status_t roctracer_disable_callback () ROCTRACER_VERSION_4_1

    Disable runtime API callback for all operations of all domains.

3.5.1 Detailed Description

ROC tracer provides support for runtime API callbacks and activity records logging. The API callbacks provide the API calls arguments and are called on different phases, on enter, on exit, on kernel completion.

3.5.2 Typedef Documentation

3.5.2.1 roctracer_rtapi_callback_t

```
typedef activity_rtapi_callback_t roctracer_rtapi_callback_t
```

Runtime API callback type.

The callback that will be invoked when an enabled runtime API is called. The callback is invoked on entry and on exit.

3.5.3 Function Documentation

3.5.3.1 roctracer_disable_callback()

ROCTRACER_API roctracer_status_t roctracer_disable_callback ( )

Disable runtime API callback for all operations of all domains.

Return values

ROCTRACER_STATUS_SUCCESS	The function has been executed successfully.

3.5.3.2 roctracer_disable_domain_callback()

ROCTRACER_API roctracer_status_t roctracer_disable_domain_callback (
            activity_domain_t *domain*)

Disable runtime API callback for all operations of a domain.

Parameters

domain	The domain

Return values

ROCTRACER_STATUS_SUCCESS	The function has been executed successfully.
ROCTRACER_STATUS_ERROR_INVALID_DOMAIN_ID	domain is invalid.

3.5.3.3 roctracer_disable_op_callback()

ROCTRACER_API roctracer_status_t roctracer_disable_op_callback (
            activity_domain_t *domain,*
            uint32_t *op*)

Disable runtime API callback for a specific operation of a domain.

Parameters

domain	The domain
op	The operation in `domain`.

Return values

[ROCTRACER_STATUS_SUCCESS](#)	The function has been executed successfully.
[ROCTRACER_STATUS_ERROR_INVALID_DOMAIN_ID](#)	`domain` is invalid.
[ROCTRACER_STATUS_ERROR_INVALID_ARGUMENT](#)	`op` is invalid for `domain`.

3.5.3.4   roctracer_enable_callback()

```
ROCTRACER_API roctracer_status_t roctracer_enable_callback (
          activity_rtapi_callback_t callback,
          void * arg )
```

Enable runtime API callback for all operations of all domains.

Parameters

callback	The callback to invoke each time the operation is performed on entry and exit.
arg	Value to pass as last argument of `callback`.

Return values

[ROCTRACER_STATUS_SUCCESS](#)	The function has been executed successfully.

3.5.3.5   roctracer_enable_domain_callback()

```
ROCTRACER_API roctracer_status_t roctracer_enable_domain_callback (
          activity_domain_t domain,
          activity_rtapi_callback_t callback,
          void * arg )
```

Enable runtime API callback for all operations of a domain.

Parameters

domain	The domain
callback	The callback to invoke each time the operation is performed on entry and exit.
arg	Value to pass as last argument of `callback`.

Return values

ROCTRACER_STATUS_SUCCESS	The function has been executed successfully.
ROCTRACER_STATUS_ERROR_INVALID_DOMAIN_ID	`domain` is invalid.

3.5.3.6 roctracer_enable_op_callback()

```
ROCTRACER_API roctracer_status_t roctracer_enable_op_callback (
          activity_domain_t domain,
          uint32_t op,
          activity_rtapi_callback_t callback,
          void * arg )
```

Enable runtime API callback for a specific operation of a domain.

Parameters

domain	The domain.
op	The operation ID in `domain`.
callback	The callback to invoke each time the operation is performed on entry and exit.
arg	Value to pass as last argument of `callback`.

Return values

ROCTRACER_STATUS_SUCCESS	The function has been executed successfully.
ROCTRACER_STATUS_ERROR_INVALID_DOMAIN_ID	`domain` is invalid.
ROCTRACER_STATUS_ERROR_INVALID_ARGUMENT	`op` is invalid for `domain`.

3.6 Activity API

The activity records are asynchronously logged to the pool and can be associated with the respective API callbacks using the correlation ID. Activity API can be used to enable collecting of the records with timestamping data for API calls and the kernel submits.

Data Structures

- struct roctracer_properties_t

    Memory pool properties.

Typedefs

- typedef activity_record_t roctracer_record_t

    Activity record.
- typedef void(∗ roctracer_allocator_t) (char ∗∗ptr, size_t size, void ∗arg)

    Memory pool allocator callback.
- typedef void(∗ roctracer_buffer_callback_t) (const char ∗begin, const char ∗end, void ∗arg)

    Memory pool buffer callback.
- typedef void roctracer_pool_t

    Tracer memory pool type.

Functions

- ROCTRACER_API roctracer_status_t roctracer_next_record (const activity_record_t ∗record, const activity_↩
    record_t ∗∗next) ROCTRACER_VERSION_4_1

    Get a pointer to the next activity record.
- ROCTRACER_API roctracer_status_t roctracer_open_pool_expl (const roctracer_properties_t ∗properties,
    roctracer_pool_t ∗∗pool) ROCTRACER_VERSION_4_1

    Create tracer memory pool.
- ROCTRACER_API   roctracer_status_t   roctracer_open_pool   (const   roctracer_properties_t   ∗properties)
    ROCTRACER_VERSION_4_1

    Create tracer memory pool.
- ROCTRACER_API roctracer_status_t roctracer_close_pool_expl (roctracer_pool_t ∗pool) ROCTRACER_VERSION_4_1

    Close tracer memory pool.
- ROCTRACER_API roctracer_status_t roctracer_close_pool () ROCTRACER_VERSION_4_1

    Close default tracer memory pool, if defined, and set to undefined.
- ROCTRACER_API roctracer_pool_t ∗ roctracer_default_pool_expl (roctracer_pool_t ∗pool) ROCTRACER_VERSION_4_1

    Query and set the default memory pool.
- ROCTRACER_API roctracer_pool_t ∗ roctracer_default_pool () ROCTRACER_VERSION_4_1

    Query the current default memory pool.
- ROCTRACER_API roctracer_status_t roctracer_enable_op_activity_expl (activity_domain_t domain, uint32_t op,
    roctracer_pool_t ∗pool) ROCTRACER_VERSION_4_1

    Enable activity record logging for a specified operation of a domain providing a memory pool.

- ROCTRACER_API roctracer_status_t roctracer_enable_op_activity (activity_domain_t domain, uint32_t op) ROCTRACER_VERSION_4_1

    Enable activity record logging for a specified operation of a domain using the default memory pool.
- ROCTRACER_API roctracer_status_t roctracer_enable_domain_activity_expl (activity_domain_t domain, roctracer_pool_t ∗pool) ROCTRACER_VERSION_4_1

    Enable activity record logging for all operations of a domain providing a memory pool.
- ROCTRACER_API roctracer_status_t roctracer_enable_domain_activity (activity_domain_t domain) ROCTRACER_VERSION_4_1

    Enable activity record logging for all operations of a domain using the default memory pool.
- ROCTRACER_API roctracer_status_t roctracer_enable_activity_expl (roctracer_pool_t ∗pool) ROCTRACER_VERSION_4_1

    Enable activity record logging for all operations of all domains providing a memory pool.
- ROCTRACER_API roctracer_status_t roctracer_enable_activity () ROCTRACER_VERSION_4_1

    Enable activity record logging for all operations of all domains using the default memory pool.
- ROCTRACER_API roctracer_status_t roctracer_disable_op_activity (activity_domain_t domain, uint32_t op) ROCTRACER_VERSION_4_1

    Disable activity record logging for a specified operation of a domain.
- ROCTRACER_API roctracer_status_t roctracer_disable_domain_activity (activity_domain_t domain) ROCTRACER_VERSION_4_1

    Disable activity record logging for all operations of a domain.
- ROCTRACER_API roctracer_status_t roctracer_disable_activity () ROCTRACER_VERSION_4_1

    Disable activity record logging for all operations of all domains.
- ROCTRACER_API roctracer_status_t roctracer_flush_activity_expl (roctracer_pool_t ∗pool) ROCTRACER_VERSION_4_1

    Flush available activity records for a memory pool.
- ROCTRACER_API roctracer_status_t roctracer_flush_activity () ROCTRACER_VERSION_4_1

    Flush available activity records for the default memory pool.

3.6.1 Detailed Description

The activity records are asynchronously logged to the pool and can be associated with the respective API callbacks using the correlation ID. Activity API can be used to enable collecting of the records with timestamping data for API calls and the kernel submits.

3.6.2 Typedef Documentation

3.6.2.1 roctracer_allocator_t

```
typedef void(* roctracer_allocator_t) (char **ptr, size_t size, void *arg)
```

Memory pool allocator callback.

If ∗ptr is NULL, then allocate memory of size bytes and save address in ∗ptr.

If ∗ptr is non-NULL and size is non-0, then reallocate the memory at ∗ptr with size size and save the address in ∗ptr. The memory will have been allocated by the same callback.

If ∗ptr is non-NULL and size is 0, then deallocate the memory at ∗ptr. The memory will have been allocated by the same callback.

size is the size of the memory allocation or reallocation, or 0 if deallocating.

arg Argument provided in the roctracer_properties_t passed to the roctracer_open_pool function.

3.6.2.2 roctracer_buffer_callback_t

```
typedef void(* roctracer_buffer_callback_t) (const char *begin, const char *end, void *arg)
```

Memory pool buffer callback.

The callback that will be invoked when a memory pool buffer becomes full or is flushed.

`begin` pointer to first entry entry in the buffer.

`end` pointer to one past the end entry in the buffer.

`arg` the argument specified when the callback was defined.

3.6.2.3 roctracer_pool_t

```
typedef void roctracer_pool_t
```

Tracer memory pool type.

3.6.2.4 roctracer_record_t

```
typedef activity_record_t roctracer_record_t
```

Activity record.

Asynchronous activity events generate activity records.

3.6.3 Function Documentation

3.6.3.1 roctracer_close_pool()

```
ROCTRACER_API roctracer_status_t roctracer_close_pool ( )
```

Close default tracer memory pool, if defined, and set to undefined.

All enabled activities that use the pool must have completed writing to the pool, before deleting the pool. Deleting a pool automatically disables any activities that specify the pool, and flushes it.

Return values

ROCTRACER_STATUS_SUCCESS	The function has been executed successfully or there is no default pool.

3.6.3.2 roctracer_close_pool_expl()

ROCTRACER_API roctracer_status_t roctracer_close_pool_expl (
              roctracer_pool_t * *pool*)

Close tracer memory pool.

All enabled activities that use the pool must have completed writing to the pool, before deleting the pool. Deleting a pool automatically disables any activities that specify the pool, and flushes it.

Parameters

in	*pool*	Memory pool to close. If NULL, the default memory pool is closed if defined. The default memory pool is set to undefined if closed.

Return values

ROCTRACER_STATUS_SUCCESS	The function has been executed successfully or pool was NULL and there is no default pool.

3.6.3.3 roctracer_default_pool()

ROCTRACER_API roctracer_pool_t* roctracer_default_pool ( )

Query the current default memory pool.

Returns

Return the current default memory pool, or NULL is none is defined.

3.6.3.4 roctracer_default_pool_expl()

ROCTRACER_API roctracer_pool_t* roctracer_default_pool_expl (
              roctracer_pool_t * *pool*)

Query and set the default memory pool.

Parameters

in	*pool*	If not NULL, change the current default pool to `pool`. If NULL, the default pool is not changed.

Returns

Return the current default memory pool before any change, or NULL if none is defined.

3.6.3.5 roctracer_disable_activity()

ROCTRACER_API roctracer_status_t roctracer_disable_activity ( )

Disable activity record logging for all operations of all domains.

Parameters

in	*op*	The activity operation ID in `domain`.

Return values

ROCTRACER_STATUS_SUCCESS	The function has been executed successfully.

3.6.3.6 roctracer_disable_domain_activity()

ROCTRACER_API roctracer_status_t roctracer_disable_domain_activity (
            activity_domain_t *domain*)

Disable activity record logging for all operations of a domain.

Parameters

in	*domain*	The domain.

Return values

ROCTRACER_STATUS_SUCCESS	The function has been executed successfully.

3.6.3.7 roctracer_disable_op_activity()

```
ROCTRACER_API roctracer_status_t roctracer_disable_op_activity (
            activity_domain_t domain,
            uint32_t op )
```

Disable activity record logging for a specified operation of a domain.

Parameters

in	*domain*	The domain.
in	*op*	The activity operation ID in `domain`.

Return values

ROCTRACER_STATUS_SUCCESS	The function has been executed successfully.

3.6.3.8 roctracer_enable_activity()

```
ROCTRACER_API roctracer_status_t roctracer_enable_activity ( )
```

Enable activity record logging for all operations of all domains using the default memory pool.

Return values

ROCTRACER_STATUS_SUCCESS	The function has been executed successfully.
ROCTRACER_STATUS_ERROR	No default pool is defined.

3.6.3.9 roctracer_enable_activity_expl()

```
ROCTRACER_API roctracer_status_t roctracer_enable_activity_expl (
            roctracer_pool_t * pool )
```

Enable activity record logging for all operations of all domains providing a memory pool.

Parameters

in	*pool*	The memory pool to write the activity record. If NULL, use the default memory pool.

Return values

ROCTRACER_STATUS_SUCCESS	The function has been executed successfully.
ROCTRACER_STATUS_ERROR	`pool` is NULL and no default pool is defined.

3.6.3.10   roctracer_enable_domain_activity()

ROCTRACER_API roctracer_status_t roctracer_enable_domain_activity (
            activity_domain_t *domain*)

Enable activity record logging for all operations of a domain using the default memory pool.

Parameters

in	*domain*	The domain.

Return values

ROCTRACER_STATUS_SUCCESS	The function has been executed successfully.
ROCTRACER_STATUS_ERROR	No default pool is defined.

3.6.3.11   roctracer_enable_domain_activity_expl()

ROCTRACER_API roctracer_status_t roctracer_enable_domain_activity_expl (
            activity_domain_t *domain,*
            roctracer_pool_t * *pool*)

Enable activity record logging for all operations of a domain providing a memory pool.

Parameters

in	*domain*	The domain.
in	*pool*	The memory pool to write the activity record. If NULL, use the default memory pool.

Return values

ROCTRACER_STATUS_SUCCESS	The function has been executed successfully.
ROCTRACER_STATUS_ERROR	`pool` is NULL and no default pool is defined.

3.6.3.12 roctracer_enable_op_activity()

ROCTRACER_API roctracer_status_t roctracer_enable_op_activity (
            activity_domain_t *domain,*
            uint32_t *op*)

Enable activity record logging for a specified operation of a domain using the default memory pool.

Parameters

in	*domain*	The domain.
in	*op*	The activity operation ID in `domain`.

Return values

ROCTRACER_STATUS_SUCCESS	The function has been executed successfully.
ROCTRACER_STATUS_ERROR	No default pool is defined.

3.6.3.13 roctracer_enable_op_activity_expl()

ROCTRACER_API roctracer_status_t roctracer_enable_op_activity_expl (
            activity_domain_t *domain,*
            uint32_t *op,*
            roctracer_pool_t * *pool*)

Enable activity record logging for a specified operation of a domain providing a memory pool.

Parameters

in	*domain*	The domain.
in	*op*	The activity operation ID in `domain`.
in	*pool*	The memory pool to write the activity record. If NULL, use the default memory pool.

Return values

ROCTRACER_STATUS_SUCCESS	The function has been executed successfully.
ROCTRACER_STATUS_ERROR	`pool` is NULL and no default pool is defined.

3.6.3.14 roctracer_flush_activity()

ROCTRACER_API roctracer_status_t roctracer_flush_activity ( )

Flush available activity records for the default memory pool.

If flushing encounters an activity record still being written, flushing stops. Use a subsequent flush when the record has completed being written to resume the flush.

Return values

ROCTRACER_STATUS_SUCCESS	The function has been executed successfully.

3.6.3.15 roctracer_flush_activity_expl()

ROCTRACER_API roctracer_status_t roctracer_flush_activity_expl (
            roctracer_pool_t * *pool*)

Flush available activity records for a memory pool.

If flushing encounters an activity record still being written, flushing stops. Use a subsequent flush when the record has completed being written to resume the flush.

Parameters

in	*pool*	The memory pool to flush. If NULL, flushes the default memory pool.

Return values

ROCTRACER_STATUS_SUCCESS	The function has been executed successfully.

3.6.3.16 roctracer_next_record()

ROCTRACER_API roctracer_status_t roctracer_next_record (
            const activity_record_t * *record,*
            const activity_record_t ** *next*)

Get a pointer to the next activity record.

A memory pool generates buffers that contain multiple activity records. This function steps to the next activity record.

Parameters

in	*record*	Pointer to ac activity record in a memory pool buffer.
out	*next*	Pointer to the following activity record in the memory pool buffer.

Return values

ROCTRACER_STATUS_SUCCESS	The function has been executed successfully.

3.6.3.17 roctracer_open_pool()

ROCTRACER_API roctracer_status_t roctracer_open_pool (
            const roctracer_properties_t * *properties*)

Create tracer memory pool.

Sets the default memory pool to the created pool if not already defined. Otherwise, return an error.

Parameters

in	*properties*	Tracer memory pool properties.

Return values

ROCTRACER_STATUS_SUCCESS	The function has been executed successfully.
ROCTRACER_STATUS_ERROR_DEFAULT_POOL↩ _ALREADY_DEFINED	The default pool is already defined. Unable to create the pool.
ROCTRACER_STATUS_ERROR_MEMORY_ALLO↩ CATION	Unable to allocate memory for the `pool`. Unable to create the pool.

3.6.3.18 roctracer_open_pool_expl()

ROCTRACER_API roctracer_status_t roctracer_open_pool_expl (
            const roctracer_properties_t * *properties,*
            roctracer_pool_t ** *pool*)

Create tracer memory pool.

If `pool` is not NULL, returns the created memory pool. Does not change the default memory pool.

If `pool` is NULL, sets the default memory pool to the created pool if not already defined. Otherwise, return an error.

Parameters

in	*properties*	Tracer memory pool properties.
out	*pool*	Tracer memory pool created if not NULL.

Return values

ROCTRACER_STATUS_SUCCESS	The function has been executed successfully.
ROCTRACER_STATUS_ERROR_DEFAULT_POOL↩_ALREADY_DEFINED	`pool` is NULL and the default pool is already defined. Unable to create the pool.
ROCTRACER_STATUS_ERROR_MEMORY_ALLO↩CATION	Unable to allocate memory for the `pool`. Unable to create the pool.

3.7 Timestamp Operations

Functions

- ROCTRACER_API roctracer_status_t roctracer_get_timestamp (roctracer_timestamp_t ∗timestamp) ROCTRACER_VERSION_4_1

  Get the system clock timestamp.

3.7.1 Detailed Description

3.7.2 Function Documentation

3.7.2.1 roctracer_get_timestamp()

```
ROCTRACER_API roctracer_status_t roctracer_get_timestamp (
            roctracer_timestamp_t * timestamp )
```

Get the system clock timestamp.

Parameters

out	*timestamp*	The system clock timestamp in nano seconds.

Return values

ROCTRACER_STATUS_SUCCESS	The function has been executed successfully.

Chapter 4

Data Structure Documentation

4.1 roctracer_properties_t Struct Reference

Memory pool properties.

```
#include <roctracer.h>
```

Data Fields

- uint32_t mode

    ROC Tracer mode.
- size_t buffer_size

    Size of buffer in bytes.
- roctracer_allocator_t alloc_fun

    The allocator function to use to allocate and deallocate the buffer.
- void ∗ alloc_arg

    The argument to pass when invoking the `alloc_fun` *allocator.*
- roctracer_buffer_callback_t buffer_callback_fun

    The function to call when a buffer becomes full or is flushed.
- void ∗ buffer_callback_arg

    The argument to pass when invoking the `buffer_callback_fun` *callback.*

4.1.1 Detailed Description

Memory pool properties.

Defines the properties when a tracer memory pool is created.

4.1.2 Field Documentation

4.1.2.1 alloc_arg

`void* roctracer_properties_t::alloc_arg`

The argument to pass when invoking the `alloc_fun` allocator.

4.1.2.2 alloc_fun

[`roctracer_allocator_t`](#) `roctracer_properties_t::alloc_fun`

The allocator function to use to allocate and deallocate the buffer.

If NULL then `malloc`, `realloc`, and `free` are used.

4.1.2.3 buffer_callback_arg

`void* roctracer_properties_t::buffer_callback_arg`

The argument to pass when invoking the `buffer_callback_fun` callback.

4.1.2.4 buffer_callback_fun

[`roctracer_buffer_callback_t`](#) `roctracer_properties_t::buffer_callback_fun`

The function to call when a buffer becomes full or is flushed.

4.1.2.5 buffer_size

`size_t roctracer_properties_t::buffer_size`

Size of buffer in bytes.

4.1.2.6 mode

`uint32_t roctracer_properties_t::mode`

ROC Tracer mode.

The documentation for this struct was generated from the following file:

- /long_pathname_so_that_rpms_can_package_the_debug_info/src/roctracer/inc/[roctracer.h](#)

Chapter 5

File Documentation

5.1 /long_pathname_so_that_rpms_can_package_the_debug_↩ info/src/roctracer/inc/roctracer.h File Reference

```
#include <stdint.h>
#include <stddef.h>
#include <ext/prof_protocol.h>
```
Include dependency graph for roctracer.h:
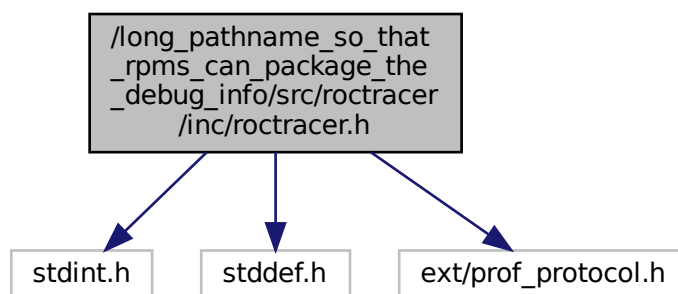
Data Structures

- struct roctracer_properties_t

     Memory pool properties.

Macros

- #define ROCTRACER_CALL
- #define ROCTRACER_EXPORT ROCTRACER_EXPORT_DECORATOR ROCTRACER_CALL
- #define ROCTRACER_IMPORT ROCTRACER_IMPORT_DECORATOR ROCTRACER_CALL
- #define ROCTRACER_API ROCTRACER_IMPORT
- #define ROCTRACER_VERSION_4_1

    The function was introduced in version 4.1 of the interface and has the symbol version string of `"ROCTRACER_4.1"`*.*
- #define ROCTRACER_VERSION_MAJOR 4

    The major version of the interface as a macro so it can be used by the preprocessor.
- #define ROCTRACER_VERSION_MINOR 1

    The minor version of the interface as a macro so it can be used by the preprocessor.

Typedefs

- typedef activity_domain_t roctracer_domain_t

    Enumeration of domains that can be traced.
- typedef activity_rtapi_callback_t roctracer_rtapi_callback_t

    Runtime API callback type.
- typedef activity_record_t roctracer_record_t

    Activity record.
- typedef void(∗ roctracer_allocator_t) (char ∗∗ptr, size_t size, void ∗arg)

    Memory pool allocator callback.
- typedef void(∗ roctracer_buffer_callback_t) (const char ∗begin, const char ∗end, void ∗arg)

    Memory pool buffer callback.
- typedef void roctracer_pool_t

    Tracer memory pool type.

Enumerations

- enum roctracer_status_t {
  ROCTRACER_STATUS_SUCCESS = 0, ROCTRACER_STATUS_ERROR = -1, ROCTRACER_STATUS_ERROR_INVALID_DOM
  = -2, ROCTRACER_STATUS_ERROR_INVALID_ARGUMENT = -3,
  ROCTRACER_STATUS_ERROR_DEFAULT_POOL_UNDEFINED = -4, ROCTRACER_STATUS_ERROR_DEFAULT_POOL_ALL
  = -5, ROCTRACER_STATUS_ERROR_MEMORY_ALLOCATION = -6, ROCTRACER_STATUS_ERROR_MISMATCHED_EXTER
  = -7,
  ROCTRACER_STATUS_UNINIT = 2, ROCTRACER_STATUS_BREAK = 3, ROCTRACER_STATUS_BAD_DOMAIN
  = ROCTRACER_STATUS_ERROR_INVALID_DOMAIN_ID, ROCTRACER_STATUS_BAD_PARAMETER =
  ROCTRACER_STATUS_ERROR_INVALID_ARGUMENT,
  ROCTRACER_STATUS_HIP_API_ERR = 6, ROCTRACER_STATUS_HIP_OPS_ERR = 7, ROCTRACER_STATUS_HCC_OPS_E
  = ROCTRACER_STATUS_HIP_OPS_ERR, ROCTRACER_STATUS_HSA_ERR = 7,
  ROCTRACER_STATUS_ROCTX_ERR = 8 }

    ROC Tracer API status codes.

Functions

- ROCTRACER_API uint32_t roctracer_version_major () ROCTRACER_VERSION_4_1

    Query the major version of the installed library.

- ROCTRACER_API uint32_t roctracer_version_minor () ROCTRACER_VERSION_4_1

    Query the minor version of the installed library.

- ROCTRACER_API const char ∗ roctracer_error_string () ROCTRACER_VERSION_4_1

    Query the textual description of the last error for the current thread.

- ROCTRACER_API const char ∗ roctracer_op_string (uint32_t domain, uint32_t op, uint32_t kind) ROCTRACER_VERSION_4_1

    Query textual name of an operation of a domain.

- ROCTRACER_API roctracer_status_t roctracer_op_code (uint32_t domain, const char ∗str, uint32_t ∗op, uint32_t ∗kind) ROCTRACER_VERSION_4_1

    Query the operation code given a domain and the name of an operation.

- ROCTRACER_API roctracer_status_t roctracer_set_properties (roctracer_domain_t domain, void ∗properties) ROCTRACER_VERSION_4_1

    Set the properties of a domain.

- ROCTRACER_API roctracer_status_t roctracer_enable_op_callback (activity_domain_t domain, uint32_t op, activity_rtapi_callback_t callback, void ∗arg) ROCTRACER_VERSION_4_1

    Enable runtime API callback for a specific operation of a domain.

- ROCTRACER_API roctracer_status_t roctracer_enable_domain_callback (activity_domain_t domain, activity_↩ rtapi_callback_t callback, void ∗arg) ROCTRACER_VERSION_4_1

    Enable runtime API callback for all operations of a domain.

- ROCTRACER_API roctracer_status_t roctracer_enable_callback (activity_rtapi_callback_t callback, void ∗arg) ROCTRACER_VERSION_4_1

    Enable runtime API callback for all operations of all domains.

- ROCTRACER_API roctracer_status_t roctracer_disable_op_callback (activity_domain_t domain, uint32_t op) ROCTRACER_VERSION_4_1

    Disable runtime API callback for a specific operation of a domain.

- ROCTRACER_API roctracer_status_t roctracer_disable_domain_callback (activity_domain_t domain) ROCTRACER_VERSION_4_

    Disable runtime API callback for all operations of a domain.

- ROCTRACER_API roctracer_status_t roctracer_disable_callback () ROCTRACER_VERSION_4_1

    Disable runtime API callback for all operations of all domains.

- ROCTRACER_API roctracer_status_t roctracer_next_record (const activity_record_t ∗record, const activity_↩ record_t ∗∗next) ROCTRACER_VERSION_4_1

    Get a pointer to the next activity record.

- ROCTRACER_API roctracer_status_t roctracer_open_pool_expl (const roctracer_properties_t ∗properties, roctracer_pool_t ∗∗pool) ROCTRACER_VERSION_4_1

    Create tracer memory pool.

- ROCTRACER_API roctracer_status_t roctracer_open_pool (const roctracer_properties_t ∗properties) ROCTRACER_VERSION_4_1

    Create tracer memory pool.

- ROCTRACER_API roctracer_status_t roctracer_close_pool_expl (roctracer_pool_t ∗pool) ROCTRACER_VERSION_4_1

    Close tracer memory pool.

- ROCTRACER_API roctracer_status_t roctracer_close_pool () ROCTRACER_VERSION_4_1

    Close default tracer memory pool, if defined, and set to undefined.

- ROCTRACER_API roctracer_pool_t ∗ roctracer_default_pool_expl (roctracer_pool_t ∗pool) ROCTRACER_VERSION_4_1

    Query and set the default memory pool.

- ROCTRACER_API roctracer_pool_t ∗ roctracer_default_pool () ROCTRACER_VERSION_4_1

Query the current default memory pool.

- [ROCTRACER_API](#) [roctracer_status_t](#) [roctracer_enable_op_activity_expl](#) (activity_domain_t domain, uint32_t op, [roctracer_pool_t](#) ∗pool) [ROCTRACER_VERSION_4_1](#)

  Enable activity record logging for a specified operation of a domain providing a memory pool.

- [ROCTRACER_API](#) [roctracer_status_t](#) [roctracer_enable_op_activity](#) (activity_domain_t domain, uint32_t op) [ROCTRACER_VERSION_4_1](#)

  Enable activity record logging for a specified operation of a domain using the default memory pool.

- [ROCTRACER_API](#) [roctracer_status_t](#) [roctracer_enable_domain_activity_expl](#) (activity_domain_t domain, [roctracer_pool_t](#) ∗pool) [ROCTRACER_VERSION_4_1](#)

  Enable activity record logging for all operations of a domain providing a memory pool.

- [ROCTRACER_API](#) [roctracer_status_t](#) [roctracer_enable_domain_activity](#) (activity_domain_t domain) [ROCTRACER_VERSION_4_1](#)

  Enable activity record logging for all operations of a domain using the default memory pool.

- [ROCTRACER_API](#) [roctracer_status_t](#) [roctracer_enable_activity_expl](#) ([roctracer_pool_t](#) ∗pool) [ROCTRACER_VERSION_4_1](#)

  Enable activity record logging for all operations of all domains providing a memory pool.

- [ROCTRACER_API](#) [roctracer_status_t](#) [roctracer_enable_activity](#) () [ROCTRACER_VERSION_4_1](#)

  Enable activity record logging for all operations of all domains using the default memory pool.

- [ROCTRACER_API](#) [roctracer_status_t](#) [roctracer_disable_op_activity](#) (activity_domain_t domain, uint32_t op) [ROCTRACER_VERSION_4_1](#)

  Disable activity record logging for a specified operation of a domain.

- [ROCTRACER_API](#) [roctracer_status_t](#) [roctracer_disable_domain_activity](#) (activity_domain_t domain) [ROCTRACER_VERSION_4_1](#)

  Disable activity record logging for all operations of a domain.

- [ROCTRACER_API](#) [roctracer_status_t](#) [roctracer_disable_activity](#) () [ROCTRACER_VERSION_4_1](#)

  Disable activity record logging for all operations of all domains.

- [ROCTRACER_API](#) [roctracer_status_t](#) [roctracer_flush_activity_expl](#) ([roctracer_pool_t](#) ∗pool) [ROCTRACER_VERSION_4_1](#)

  Flush available activity records for a memory pool.

- [ROCTRACER_API](#) [roctracer_status_t](#) [roctracer_flush_activity](#) () [ROCTRACER_VERSION_4_1](#)

  Flush available activity records for the default memory pool.

- [ROCTRACER_API](#) [roctracer_status_t](#) [roctracer_get_timestamp](#) (roctracer_timestamp_t ∗timestamp) [ROCTRACER_VERSION_4_1](#)

  Get the system clock timestamp.

5.1.1 Detailed Description

ROCtracer API interface.

5.1.2 Macro Definition Documentation

5.1.2.1 ROCTRACER_API

```
#define ROCTRACER_API ROCTRACER_IMPORT
```

5.1.2.2 ROCTRACER_CALL

```
#define ROCTRACER_CALL
```

5.1.2.3 ROCTRACER_EXPORT

```
#define ROCTRACER_EXPORT ROCTRACER_EXPORT_DECORATOR ROCTRACER_CALL
```

5.1.2.4 ROCTRACER_IMPORT

```
#define ROCTRACER_IMPORT ROCTRACER_IMPORT_DECORATOR ROCTRACER_CALL
```

Index