

GCCTrace

Generated by Doxygen 1.8.3.1

Tue Apr 8 2014 21:43:35

Contents

1	Class Index	1
1.1	Class List	1
2	File Index	3
2.1	File List	3
3	Class Documentation	5
3.1	call_stack Struct Reference	5
3.1.1	Detailed Description	5
3.1.2	Member Data Documentation	5
3.1.2.1	frames	5
3.1.2.2	num_frames	5
3.2	stack_frame Struct Reference	5
3.2.1	Detailed Description	6
3.2.2	Member Data Documentation	6
3.2.2.1	call_site	6
3.2.2.2	this_fn	6
3.2.2.3	thread	6
3.2.2.4	time	6
3.2.2.5	used_bytes	6
4	File Documentation	7
4.1	include/gcctrace.h File Reference	7
4.1.1	Detailed Description	8
4.1.2	Typedef Documentation	8
4.1.2.1	call_stack	8
4.1.2.2	stack_frame	8
4.1.3	Function Documentation	8
4.1.3.1	__cyg_profile_func_enter	8
4.1.3.2	__cyg_profile_func_exit	8
4.1.3.3	_gcc_trace_clone_current_call_stack	8
4.1.3.4	_gcc_trace_free_call_stack	9

4.1.3.5	_gcc_trace_print_call_stack	9
4.1.3.6	calloc	9
4.1.3.7	free	9
4.1.3.8	malloc	9
4.1.3.9	realloc	10

Index	10
--------------	-----------

Chapter 1

Class Index

1.1 Class List

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

call_stack	Full calling stack	5
stack_frame	Stack frame structure	5

Chapter 2

File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

include/ gcctrace.h	
Main file to include gcctrace's functionalities	7

Chapter 3

Class Documentation

3.1 `call_stack` Struct Reference

Full calling stack.

```
#include <gcctrace.h>
```

Public Attributes

- unsigned int [num_frames](#)
- [stack_frame](#) * [frames](#)

3.1.1 Detailed Description

Full calling stack.

Full calling stack

3.1.2 Member Data Documentation

3.1.2.1 `stack_frame*` `call_stack::frames`

Stack frames array

3.1.2.2 unsigned int `call_stack::num_frames`

How deep is the stack

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

- [include/gcctrace.h](#)

3.2 `stack_frame` Struct Reference

Stack frame structure.

```
#include <gcctrace.h>
```

Public Attributes

- unsigned long int [time](#)
- unsigned long int [thread](#)
- unsigned long int [used_bytes](#)
- void * [this_fn](#)
- void * [call_site](#)

3.2.1 Detailed Description

Stack frame structure.

Stack frame containing the traced data

3.2.2 Member Data Documentation

3.2.2.1 void* `stack_frame::call_site`

Place in the code where `this_fn` was called

3.2.2.2 void* `stack_frame::this_fn`

Pointer to the invoked function

3.2.2.3 unsigned long int `stack_frame::thread`

Thread ID

3.2.2.4 unsigned long int `stack_frame::time`

Timestamp

3.2.2.5 unsigned long int `stack_frame::used_bytes`

Allocated memory in bytes

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

- [include/gcctrace.h](#)

Chapter 4

File Documentation

4.1 include/gcctrace.h File Reference

Main file to include gcctrace's functionalities.

```
#include <stdlib.h>
```

Classes

- struct [stack_frame](#)
Stack frame structure.
- struct [call_stack](#)
Full calling stack.

Typedefs

- typedef struct [stack_frame](#) [stack_frame](#)
Stack frame structure.
- typedef struct [call_stack](#) [call_stack](#)
Full calling stack.

Functions

- void [__cyg_profile_func_enter](#) (void *this_fn, void *call_site) [__attribute__\(\(no_instrument_function\)\)](#)
enter function
- void [__cyg_profile_func_exit](#) (void *this_fn, void *call_site) [__attribute__\(\(no_instrument_function\)\)](#)
exit function
- void * [malloc](#) (size_t size) [__attribute__\(\(no_instrument_function\)\)](#)
- void [free](#) (void *ptr) [__attribute__\(\(no_instrument_function\)\)](#)
- void * [calloc](#) (size_t nmemb, size_t size) [__attribute__\(\(no_instrument_function\)\)](#)
- void * [realloc](#) (void *ptr, size_t size) [__attribute__\(\(no_instrument_function\)\)](#)
- void [_gcc_trace_clone_current_call_stack](#) ([call_stack](#) *stack)
- void [_gcc_trace_free_call_stack](#) ([call_stack](#) *stack)
- void [_gcc_trace_print_call_stack](#) ([call_stack](#) *stack)

4.1.1 Detailed Description

Main file to include gcctrace's functionalities.

Author

Renato Grottesi

Date

7 Apr 2014

4.1.2 Typedef Documentation

4.1.2.1 typedef struct call_stack call_stack

Full calling stack.

Full calling stack

4.1.2.2 typedef struct stack_frame stack_frame

Stack frame structure.

Stack frame containing the traced data

4.1.3 Function Documentation

4.1.3.1 void __cyg_profile_func_enter (void * *this_fn*, void * *call_site*)

enter function

This function will be called before any other function can start

Parameters

<i>this_fn</i>	Function getting called
<i>call_site</i>	Place in the source code where <code>this_func</code> is getting called

4.1.3.2 void __cyg_profile_func_exit (void * *this_fn*, void * *call_site*)

exit function

This function will be called before any other function returns

Parameters

<i>this_fn</i>	Function that was called
<i>call_site</i>	Place in the source code where <code>this_func</code> was called

4.1.3.3 void _gcc_trace_clone_current_call_stack (call_stack * *stack*)

Copy the current call stack inside the input parameter. This function allocates stack's internal memory. Please call `_gcc_trace_free_call_stack` to free the internal memory.

Parameters

<i>stack</i>	The call stack object where to clone the current call stack
--------------	---

See Also

[_gcc_trace_free_call_stack](#)

4.1.3.4 void `_gcc_trace_free_call_stack (call_stack * stack)`

Free the internal memory allocated by `_gcc_trace_clone_current_call_stack`

Parameters

<i>stack</i>	The call stack object to free
--------------	-------------------------------

See Also

[_gcc_trace_clone_current_call_stack](#)

4.1.3.5 void `_gcc_trace_print_call_stack (call_stack * stack)`

Prints a call stack cloned by `_gcc_trace_clone_current_call_stack`.

Parameters

<i>stack</i>	The stack to print in stderr
--------------	------------------------------

See Also

[_gcc_trace_clone_current_call_stack](#)

4.1.3.6 void* `calloc (size_t nmemb, size_t size)`

libc calloc function wrapper

Parameters

<i>nmemb</i>	Numer of members to allocate
<i>size</i>	How many bytes to allocate

4.1.3.7 void `free (void * ptr)`

libc free function wrapper

Parameters

<i>ptr</i>	The pointer to free
------------	---------------------

4.1.3.8 void* `malloc (size_t size)`

libc malloc function wrapper

Parameters

<i>size</i>	How many bytes to allocate
-------------	----------------------------

4.1.3.9 void* realloc (void * *ptr*, size_t *size*)

libc realloc function wrapper

Parameters

<i>ptr</i>	The pointer to free
<i>size</i>	How many bytes to allocate

Index

- [__cyg_profile_func_enter](#)
gcctrace.h, [8](#)
 - [__cyg_profile_func_exit](#)
gcctrace.h, [8](#)
 - [_gcc_trace_clone_current_call_stack](#)
gcctrace.h, [8](#)
 - [_gcc_trace_free_call_stack](#)
gcctrace.h, [9](#)
 - [_gcc_trace_print_call_stack](#)
gcctrace.h, [9](#)
- [call_site](#)
 - [stack_frame](#), [6](#)
- [call_stack](#), [5](#)
 - [frames](#), [5](#)
 - gcctrace.h, [8](#)
 - [num_frames](#), [5](#)
- [calloc](#)
 - gcctrace.h, [9](#)
- [frames](#)
 - [call_stack](#), [5](#)
- [free](#)
 - gcctrace.h, [9](#)
- [gcctrace.h](#)
 - [__cyg_profile_func_enter](#), [8](#)
 - [__cyg_profile_func_exit](#), [8](#)
 - [_gcc_trace_clone_current_call_stack](#), [8](#)
 - [_gcc_trace_free_call_stack](#), [9](#)
 - [_gcc_trace_print_call_stack](#), [9](#)
 - [call_stack](#), [8](#)
 - [calloc](#), [9](#)
 - [free](#), [9](#)
 - [malloc](#), [9](#)
 - [realloc](#), [10](#)
 - [stack_frame](#), [8](#)
- [include/gcctrace.h](#), [7](#)
- [malloc](#)
 - gcctrace.h, [9](#)
- [num_frames](#)
 - [call_stack](#), [5](#)
- [realloc](#)
 - gcctrace.h, [10](#)
- [stack_frame](#), [5](#)
 - [call_site](#), [6](#)
 - gcctrace.h, [8](#)
 - [this_fn](#), [6](#)
 - [thread](#), [6](#)
 - [time](#), [6](#)
 - [used_bytes](#), [6](#)
- [this_fn](#)
 - [stack_frame](#), [6](#)
- [thread](#)
 - [stack_frame](#), [6](#)
- [time](#)
 - [stack_frame](#), [6](#)
- [used_bytes](#)
 - [stack_frame](#), [6](#)