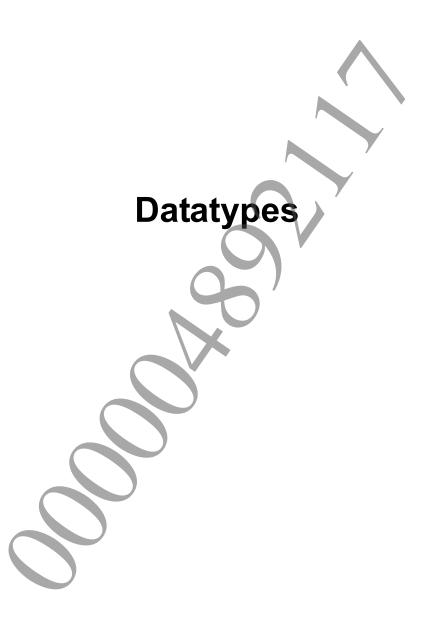


© 2015 Sony Computer Entertainment Inc. All Rights Reserved. SCE Confidential

Table of Contents

Datatypes	3
SceRazorCpuUserMarkerTracePacket	4
SceRazorCpuActivityMonitorPacket	5
Functions	6
scePerfArmPmonReset	7
scePerfArmPmonSelectEvent	8
scePerfArmPmonStart	g
scePerfArmPmonStop	10
scePerfArmPmonGetCounterValue	11
scePerfArmPmonSetCounterValue	12
scePerfArmPmonSoftwareIncrement	
scePerfGetTimebaseValue	14
scePerfGetTimebaseFrequency	15
sceRazorCpuGetActivityMonitorTraceBuffer	
sceRazorCpuGetUserMarkerTraceBuffer	17
sceRazorCpuPushMarker	18
sceRazorCpuPushMarkerWithHud	
sceRazorCpuPopMarker	20
sceRazorCpuStartActivityMonitor	21
sceRazorCpuStartUserMarkerTrace sceRazorCpuStopActivityMonitor	22
sceRazorCpuStopUserMarkerTrace	
sceRazorCpuStartCapture	25
sceRazorCpuStopCapture	26
sceRazorCpuSync	
sceRazorCpulsCapturing	
Constants	29
Return Codes	30
Define Summary	31



SceRazorCpuUserMarkerTracePacket

A CPU user marker trace packet

Definition

Members

header The Razor identifier

threadId Thread ID

stackLevel Position of user marker in hierarchy

color Color of the user marker for display on HUD

timestamp of the marker

Description

Packet format for HUD user markers.

SceRazorCpuActivityMonitorPacket

A CPU activity monitor trace packet

Definition

Members

systemSystem activity time in nanosecondsidleIdle thread time in nanosecondsuserUser thread time in nanoseconds

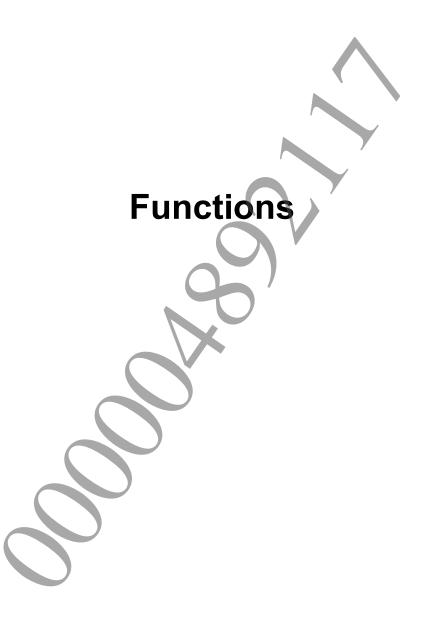
frameIndex Index for this timeframe

cpuId CPU core ID

Description

Packet format for CPU activity monitor events.





scePerfArmPmonReset

Reset event counter and cycle counter

Definition

```
#include <libperf.h>
int scePerfArmPmonReset(
        SceUID threadId
);
```

Arguments

threadId Thread ID

Return Values

Returns SCE OK for normal termination. Returns an error code (a negative value) for errors.

Description

This function resets the event counter and cycle counter. Calling this function resets all counters in applicable threads.

Specify the thread ID to threadId. The following macro definitions can be used as special thread IDs.

Valu	ıe						Description	
SCE_	PERF	ARM	_PMON_	THREAD	_ID_	SELF	Self thread	
SCE	PERF	ARM	PMON	THREAD	ID	ALL	All existing thre	eads



scePerfArmPmonSelectEvent

Select Performance Monitor event

Definition

Arguments

```
threadId Thread ID counter Code eventCode Event code
```

Return Values

Returns SCE OK for normal termination.

Returns the following error code (a negative value) for errors.

Valu	ıe				Description
SCE	PERF_	_ERROR_	INVALID	ARGUMENT	Invalid argument

Description

This function selects the Performance Monitor event.

Specify the thread ID to threadId. The following macro definitions can be used as special thread IDs.

I	Valu	e						Description
Ī	SCE	PERF	ARM	PMON	THREAD	ID	SELF	Self thread
Ī	SCE	PERF	ARM	PMON	THREAD	ID	ALL	All existing threads

Specify the counter code to *counter*. There are six event counters whose values can be specified from 0 to 5.

Specify the event code to event Code. Refer to the "ARM Architecture Reference Manual ARMv7-A and ARMv7-R edition" and the "Cortex-A9 Technical Reference Manual Revision: r3p0" for event details.

scePerfArmPmonStart

Start measuring

Definition

```
#include <libperf.h>
int scePerfArmPmonStart(
        SceUID threadId
);
```

Arguments

threadId Thread ID

Return Values

Returns SCE OK for normal termination. Returns an error code (a negative value) for errors.

Description

This function starts measuring performance.

Performance measurement begins when this function is called and continues until scePerfArmPmonStop() is called.

Specify the thread ID to threadId. The following macro definitions can be used as special thread IDs.

Value		Description
		_ID_SELF Self thread
SCE_PERF_A	ARM_PMON_THREAD	_ID_ALL All existing threads



scePerfArmPmonStop

Stop measuring

Definition

```
#include <libperf.h>
\verb"int scePerfArmPmonStop" (
         SceUID threadId
);
```

Arguments

threadId Thread ID

Return Values

Returns SCE OK for normal termination. Returns an error code (a negative value) for errors.

Description

This function stops measuring performance.

Specify the thread ID to threadId. The following macro definitions can be used as special thread IDs.

Valu	e						Description
							Self thread
SCE	PERF	ARM	PMON	THREAD	ID	ALL	All existing threads

scePerfArmPmonGetCounterValue

Get counter value

Definition

Arguments

threadId Thread ID

counter Counter code

Planta to variable who

pValue Pointer to variable where counter value is stored

Return Values

Returns a value equal to or greater than 0 for normal termination. Returns the following error code (a negative value) for errors.

Valu	ıe				Description
SCE_	PERF_	_ERROR_	_INVALID_	_ARGUMENT	Invalid argument

Description

This function obtains the event counter and cycle counter values.

Specify the thread ID to threadId. The following macro definition can be used as a special thread ID. For this function, all existing threads (SCE_PERF_ARM_PMON_THREAD_ID_ALL) cannot be specified.



To *counter*, specify the counter code whose value you wish to obtain. There are six event counters whose values can be specified from 0 to 5. For the cycle counter, specify 31.

The counter value is stored in *pValue.

©SCEI

scePerfArmPmonSetCounterValue

Set counter value

Definition

```
#include <libperf.h>
int scePerfArmPmonSetCounterValue(
        SceUID threadId,
        SceUInt32 counter,
        SceUInt32 value
);
```

Arguments

threadId Thread ID Counter code counter value Counter value

Return Values

Returns SCE OK for normal termination.

Returns an error code (a negative value) for errors.

Description

This function sets the event counter and cycle counter values.

Specify the thread ID to threadId. The following macro definitions can be used as special thread IDs.

Valu							Description
							Self thread
SCE	PERF	ARM	PMON	THREAD	ID	ALL	All existing threads

To counter, specify the counter code whose value you wish to set. There are six event counters whose values can be specified from 0 to 5. For the cycle counter, specify 31.

The counter value specified to value will be set.



scePerfArmPmonSoftwareIncrement

Increment Software Increment event

Definition

Arguments

mask Bit mask for counter code

Return Values

Returns ${\tt SCE_OK}$ for normal termination.

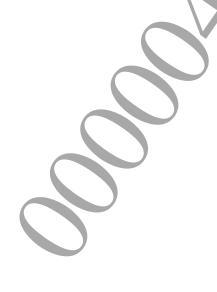
Returns the following error code (a negative value) for errors.

	Valu					Description
ı	SCE	PERF	ERROR	INVALID	ARGUMENT	Invalid argument

Description

Software Increment is an event defined in the Performance Monitor functions on the ARM Processor. This event allows the event counter value to be controlled in the software.

To *mask*, specify the bit mask of the event counter you wish to increment.



scePerfGetTimebaseValue

Get timebase value

Definition

#include <libperf.h>
SceUInt64 scePerfGetTimebaseValue(void);

Arguments

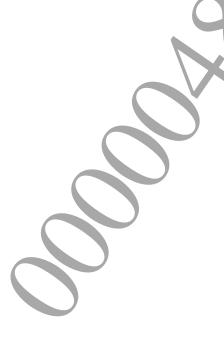
None

Return Values

Returns timebase value.

Description

This function returns the value of 48 bits timebase counter. The timebase counter is free running and can be used as a global timer. The frequency of this timebase counter can be obtained by <code>scePerfGetTimebaseFrequency()</code>.



scePerfGetTimebaseFrequency

Get timebase frequency

Definition

#include <libperf.h> SceUInt32 scePerfGetTimebaseFrequency(void);

Arguments

None

Return Values

Returns timebase frequency in MHz.

Description

This function returns the frequency of 48 bits timebase counter. The timebase counter is free running and can be used as a global timer. The value of this timebase counter can be obtained by scePerfGetTimebaseValue().



sceRazorCpuGetActivityMonitorTraceBuffer

Get the CPU activity monitor trace buffer

Definition

```
#include <libperf.h>
int sceRazorCpuGetActivityMonitorTraceBuffer (
        SceRazorCpuActivityMonitorPacket **pTrace
);
```

Arguments

pTrace Output parameter. Contains address of trace buffer

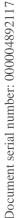
Return Values

Returns number of elements in trace buffer for normal termination. Returns the following error code (a negative value) for errors.

Valu	ıe					Description
SCE	PERF	ERROR	BAD	TRACE	DATA	Invalid trace data

Description

This function gets a pointer to the most recent activity monitor data. Internally this swaps a double buffer, so tracing continues while the buffer is being processed. This should not be called when using the Razor HUD - only when writing a customized performance HUD.



sceRazorCpuGetUserMarkerTraceBuffer

Get the HUD user marker trace buffer

Definition

Arguments

pTrace Output parameter. Contains address of trace buffer

Return Values

Returns number of elements in trace buffer for normal termination. Returns an error code (a negative value) for errors.

Description

This function gets a pointer to the most recent user marker data. Internally this swaps a double buffer, so tracing continues while the buffer is being processed. This should not be called when using the Razor HUD – only when writing a customized performance HUD.



Document serial number: 000004892117

sceRazorCpuPushMarker

Push a marker

Definition

Arguments

szLabel Label to describe the marker

Return Values

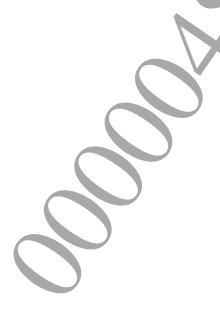
Returns 0 for normal termination.

Returns an error code (a negative value) for errors.

Description

This function pushes a marker with a specified label. A maximum of 64 markers per thread may be pushed onto the stack.

This function is deprecated. Use sceRazorCpuPushMarkerWithHud() with the flags parameter set to SCE RAZOR MARKER DISABLE HUD for equivalent behavior.



sceRazorCpuPushMarkerWithHud

Push a marker with support for the Razor HUD

Definition

Arguments

szLabel Label to describe the marker

color Color of marker on HUD

flags SCE_RAZOR_MARKER_DISABLE_HUD to disable marker on HUD

SCE_RAZOR_MARKER_ENABLE_HUD to enable on HUD

Return Values

Returns 0 for normal termination.

Returns an error code (a negative value) for errors.

Description

This function pushes a marker with a specified label. Markers are visible to the Razor host tool, the Razor HUD (if <code>flags</code> is set to <code>SCE_RAZOR_MARKER_ENABLE_HUD</code>). Users can create their own custom performance HUD by creating the trace buffers using <code>sceRazorCpuStartUserMarkerTrace()</code> and accessing the trace through <code>sceRazorCpuGetUserMarkerTraceBuffer()</code>. A maximum of 64 markers per thread may be pushed onto the stack.



sceRazorCpuPopMarker

Pop a marker

Definition

#include <libperf.h> int sceRazorCpuPopMarker(void);

Arguments

None

Return Values

Returns 0 for normal termination. Returns an error code (a negative value) for errors.

Description

This function pops a marker.



sceRazorCpuStartActivityMonitor

Assign buffer for a CPU activity monitor trace, and start tracing

Definition

```
#include <libperf.h>
int sceRazorCpuStartActivityMonitor (
        void* pBufferBase,
        SceUInt32 bufferSize
);
```

Arguments

pBufferBase Base of the buffer used to store trace events (must be 8-byte aligned) Size of the buffer (must be a multiple of 8-bytes) bufferSize

Return Values

Returns 0 for normal termination.

Returns an error code (a negative value) for errors.

Description

This function allocates a buffer and starts tracing CPU activity monitor. This can be used for creating customized Performance HUDs. This should not be called when using the Razor HUD - only when writing a customized performance HUD.

Call this once at the start of your code. Use a buffer size of at least 256KiB.



sceRazorCpuStartUserMarkerTrace

Assign buffer for user marker trace, and start tracing

Definition

```
#include <libperf.h>
int sceRazorCpuStartUserMarkerTrace (
        void* pBufferBase,
        SceUInt32 bufferSize
);
```

Arguments

pBufferBase Base of the buffer used to store trace events (must be 8-byte aligned) Size of the buffer (must be a multiple of 8-bytes) bufferSize

Return Values

Returns 0 for normal termination.

Returns an error code (a negative value) for errors.

Description

This function allocates a buffer and starts tracing user markers. User markers are created with sceRazorCpuPushMarkerWithHud. This should not be called when using the Razor HUD - only when writing a customized performance HUD.

Call this once at the start of your code. Use a buffer size of around 1-2MB.



sceRazorCpuStopActivityMonitor

Stop tracing CPU activity monitor

Definition

#include <libperf.h>
int sceRazorCpuStopActivityMonitor ();

Arguments

None

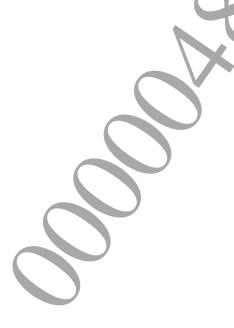
Return Values

Returns 0 for normal termination.

Returns an error code (a negative value) for errors.

Description

This function stops tracing CPU activity monitor. This should not be called when using the Razor HUD – only when writing a customized performance HUD.



sceRazorCpuStopUserMarkerTrace

Stop tracing user markers

Definition

#include <libperf.h>
int sceRazorCpuStopUserMarkerTrace ();

Arguments

None

Return Values

Returns 0 for normal termination.

Returns an error code (a negative value) for errors.

Description

This function stops tracing user markers. This should not be called when using the Razor HUD – only when writing a customized performance HUD.



sceRazorCpuStartCapture

Start Razor CPU capture

Definition

#include <libperf.h> int sceRazorCpuStartCapture(void);

Arguments

None

Return Values

Returns 0 for normal termination.

Returns an error code (a negative value) for errors.

Description

This function enables a Razor CPU capture to be started from within the source code.

Notes

Before using this function, select Start Capture in the Listen for Target Triggers option for Razor. If this setting is not performed, an error will return from this function. For details, refer to the "Performance Analysis and GPU Debugging" document.



sceRazorCpuStopCapture

Stop Razor CPU capture

Definition

#include <libperf.h> int sceRazorCpuStopCapture(void);

Arguments

None

Return Values

Returns 0 for normal termination.

Returns an error code (a negative value) for errors.

Description

This function enables a Razor CPU capture to be stopped from within the source code.

Notes

Before using this function, select Stop Capture in the Listen for Target Triggers option for Razor. If this setting is not performed, an error will return from this function. For details, refer to the "Performance Analysis and GPU Debugging" document.



sceRazorCpuSync

Synchronization point for Razor

Definition

#include <libperf.h>
int sceRazorCpuSync(void);

Arguments

None

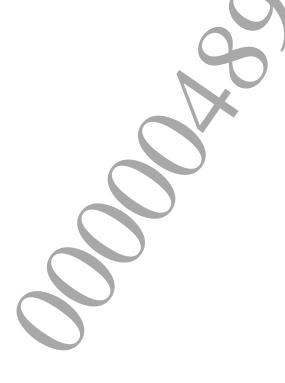
Return Values

Returns 0 for normal termination.

Returns an error code (a negative value) for errors.

Description

Call this function periodically to implement custom frame boundaries. Razor can use these events to break the timeline into frames.



sceRazorCpulsCapturing

Query if host tool is doing a CPU capture

Definition

#include <libperf.h> SceUInt32 sceRazorCpuIsCapturing(void);

Arguments

None

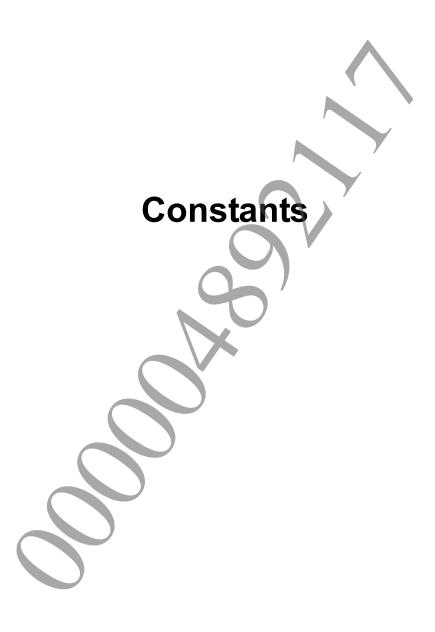
Return Values

Returns SCE RAZOR NOT_CAPTURING when not capturing. Returns SCE RAZOR CAPTURING when capturing.

Description

Call this function to determine if the host tool is currently doing a CPU capture.





Return Codes

Return codes returned by libperf module

Definition

Value	Description
SCE_PERF_ERROR_INVALID_ARGUMENT	Invalid argument
SCE_PERF_ERROR_BAD_TRACE_DATA	Invalid trace data
SCE_PERF_ERROR_POP_WITHOUT_PUSH	Attempting to pop an empty marker stack
SCE_PERF_ERROR_TOO_MANY_PUSHES	Attempting to push beyond the thread or fiber stack limit
SCE_PERF_ERROR_NOT_INITIALIZED	Module is not initialized
SCE_PERF_ERROR_ALREADY_STARTED	Cannot start capture because host-side capture is already
	started
SCE_PERF_ERROR_CANNOT_START	Cannot start capture
SCE_PERF_ERROR_ALREADY_STOPPED	Cannot stop capture because host-side capture is already
	stopped
SCE_PERF_ERROR_CANNOT_STOP	Cannot stop capture

Description

The libperf functions may return error code returned from kernel module. Refer to the "Kernel Reference" document for kernel error codes.



Define Summary

Macro defines available for use with the libperf module

Definition

Value	(Number)	Description
SCE_RAZOR_COLOR_RED	0x800000ff	Red color user marker
SCE_RAZOR_COLOR_GREEN	0x8000ff00	Green color user marker
SCE_RAZOR_COLOR_BLUE	0x80ff0000	Blue color user marker
SCE_RAZOR_COLOR_YELLOW	0x8000ffff	Yellow color user marker
SCE_RAZOR_COLOR_MAGENTA	0x80ff00ff	Magenta color user marker
SCE_RAZOR_COLOR_CYAN	0x80ffff00	Cyan color user marker
SCE_RAZOR_COLOR_WHITE	0x80ffffff	White color user marker
SCE_RAZOR_COLOR_BLACK	0x80000000	Black color user marker
SCE_RAZOR_MARKER_DISABLE_HUD	0	Disable marker on HUD
SCE_RAZOR_MARKER_ENABLE_HUD	1	Enable marker on HUD
SCE_RAZOR_NOT_CAPTURING	0	Host tool is not doing a CPU capture
SCE_RAZOR_CAPTURING	1	Host tool is doing a CPU capture

