

MPSHade

User's Manual

v1.0.1

Copyright 2006 Sun Microsystems, Inc. 901 San Antonio Road, Palo Alto, California 94303, U.S.A. All rights reserved.

This product or document is protected by copyright and distributed under licenses restricting its use, copy-

NAME

mpspixcounts - generate spix counts files for multiple applications

SYNOPSIS

mpspixcounts [**-b** *fmt*] [**-o** *outfile*] [**-s** *signal*] [**-server**] [**-shlibs**] [**-data** *fmt*]
[**--** *command*]

DESCRIPTION

The **mpspixcounts** MPShade analyzer generates one or more **spixcounts(5sh)** format files for a set of commands. The *spixcounts* files can be used with the SpixTools commands to produce detailed execu-

mpfinish utility. The **-s** switch may be especially useful in conjunction with **-server**.

-shlibs Normally, **mpspixcounts**

NAME

mpspixtracer – attach to an mpspixcounts analyzer

SYNOPSIS

mpspixtracer [**-shade** *shade-options*] [*range-options*] **-attach** *pid* [**--**] *command*

DESCRIPTION

The **mpspixtracer** MPShade tracer attaches to an existing **mpspixcounts** analyzer and starts tracttarpDESCRIPTION

NAME

mptracer – Standard MPSHade tracer

SYNOPSIS

mptracer

NAME

mpshade_intro – introduction to MPSHade library

DESCRIPTION

Combined-Trace Layer

The second MPShade library layer is called the combined-trace layer. Analyzers written to this layer do not respond to events from the tracers. Instead, they see a single stream of trace records that is time-

mpshade_load(3sh)
mpshade_trsize(3sh)
mpshade_iset_newclass(3sh)
mpshade_iset_newtype(3sh)
mpshade_iset_newop(3sh)
mpshade_iset_newcopy(3sh)
mpshade_iset_free(3sh)
mpshade_iset_addclass(3sh)
mpshade_iset_addtype(3sh)
mpshade_iset_addop(3sh)
mpshade_tset_new(3sh)
mpshade_tset_newcopy(3sh)
mpshade_tset_free(3sh)
mpshade_tset_add(3sh)
mpshade_trctl(3sh)
mpshade_setmode(3sh)

In addition, the standard tracer **mptracer(1sh)** can attach to analyzers using this layer.

FILES

\$\$SHADE	Shade installation base directory
\$\$SHADE/inc	MPSHade C header files
\$\$SHADE/lib/libmpshadea.a	MPSHade analyzer library functions
\$\$SHADE/lib/libmpshadet.a	MPSHade tracer library functions
\$\$SHADE/bin/mptracer	MPSHade standard tracer

SEE ALSO

mpshade_main(3sh)
"Introduction to Shade".
"Shade User's Manual".

NAME

mpshade_main, mpshadeuser_analyzer, mpshadeuser_combined, mpshadeuser_stdtr, mpshadeuser_tracer
– MPShade entry points

SYNOPSIS

```
int mpshadeuser_analyzer(int argc, char **argv, char **envp);  
int mpshadeuser_combined(int argc, char **argv, char **envp);  
int mpshadeuser_stdtr(int argc, char **argv, char **envp);  
int mpshadeuser_tracer(int argc, char **argv, char **envp);
```

DESCRIPTION

These are the user defined entry points for MPShade analyzers and tracers. An MPShade analyzer must supply exactly one of **mpshadeuser_analyzer()**, **mpshadeuser_combined()**, or **mpshadeuser_stdtr()** depending on the library layer used. If the analyzer uses only primitive layer functions, define **mpshadeuser_analyzer()**. If the analyzer uses combined-trace layer functions but not standard-tracer layer functions, define **mpshadeuser_combined()**. Otherwise, define **mpshadeuser_stdtr()** for analyzers

NAME

mpshade

NAME

mpshade_evt_wait – Wait for an event from an MPSHade tracer

SYNOPSIS

```
#include <mpshade.h>
```

```
mpshade
```

7de);1 Tfracer

```
μπ4.δε _e7t
```

```
μπ7t
```

environment strings for the application. If the application is run under an interpreter (eg. the dynamic loader), the *path_interp* and *addr_interp*

NAME

mpshade_evt_waitid – Wait for specific event from

SYNOPSIS

```
#include <mpshade.h>
```

```
mpshade_etype_t mpshade_evt_waitid(mpshade_id_t tracermask,  
mpshade_etype_t evtmask,  
mpshade_waitflag_t flags, mpshade_evt_t *pev)
```

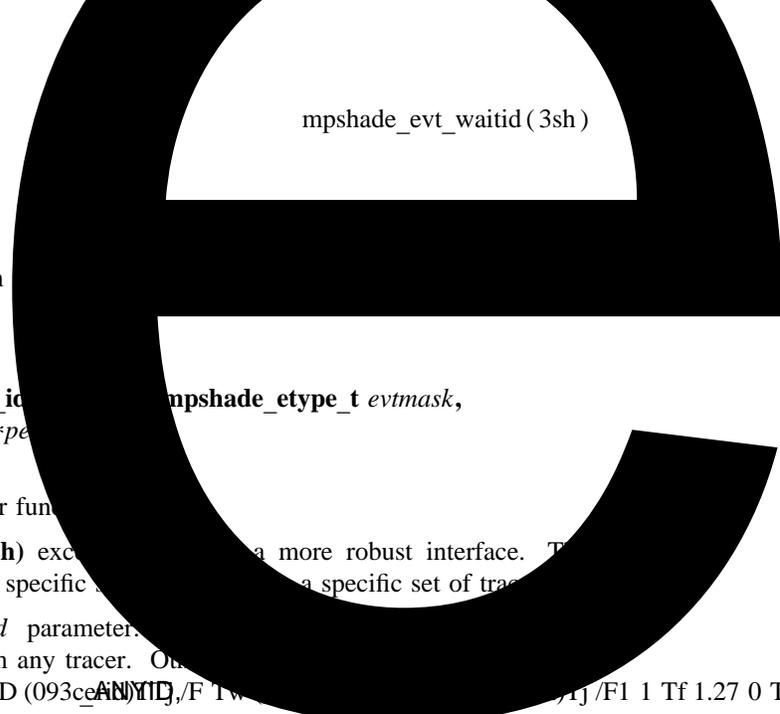
DESCRIPTION

This is an MPSHade library primitive layer analyzer function.

This function is similar to **mpshade_evt_wait(3sh)** except that it provides a more robust interface. The **mpshade_evt_waitid()** function waits for one of a specific set of events from a specific set of tracers.

The set of tracers is defined by the *tracermask* parameter.

mpshade_evt_waitid() will wait for an event from any tracer. Otherwise, **mpshade_evt_waitid()** will wait for an event from a specific tracer, and **mpshade_evt_waitid()** will wait for an event from any tracer.



NAME

mpshade_report_trace

NAME

mpshade_attach – Attach to an existing MPSHade analyzer

SYNOPSIS

```
#include <mpshade.h>
```

```
int mpshade_attach(pid_t pid);
```

DESCRIPTION

This is an MPSHade library primitive layer tracer function.

NAME

mpshade_ntracers – Get the number of MPShade tracers

SYNOPSIS

```
#include <mpshade.h>
```

```
mpshade_id_t mpshade_ntracers(void);
```

DESCRIPTION

This is an MPShade library combined-trace layer analyzer function.

Check for newly attached tracers and return the number of tracers managed by this analyzer. All IDs from zero through one less than the returned ID are guaranteed to be valid inputs to **mpshade_appinfo(3sh)**.

RETURNS

The number of MPShade tracers.

ERRORS

None.

SEE ALSO

mpshade_appinfo(3sh).

NAME

mpshade_setcallback – Setup call-back function for events

SYNOPSIS

```
#include <mpshade.h>
```

```
int mpshade_setcallback(mpshade_etype_t etype,  
                        void (*pf)(mpshade_etype_t, const mpshade_evt_t *, void *),  
                        void *pdata);
```

DESCRIPTION

This is an MPShade library combined-trace layer analyzer function.

Set up a call-back function that is invoked whenever an event of the specified type is received. Call-

NAME

mpshade_setrunflags – Switch between blocking and non-blocking tracing modes

SYNOPSIS

```
#include <mpshade.h>
```

```
int mpshade_setrunflags(mpshade_waitflag_t flags);
```

DESCRIPTION

This is an MPShade library combined-trace layer analyzer function.

This function affects the behavior of future calls to **mpshade_run(3sh)**, **mpshade_runid(3sh)**, **mpshade_step(3sh)**, and **mpshade_stepid(3sh)**. If *flags* contains **MPSHADE_NOBLOCK**, these functions immediately return zero when no tracer records are available. Otherwise, they wait until a tracer record is ready. If *flags* contains **MPSHADE_NOFINISH**, they wait indefinitely when all tracers have exited. Otherwise, they return zero when all tracers have exited.

RETURNS

Returns 0 on success, **MPSHADE_ERROR** on error.

ERRORS

Invalid flag value.

SEE ALSO

mpshade_run(3sh), mpshade_step(3sh).

NAME

mpshade_setslice – Set the number of records in an MPShade time-slice

SYNOPSIS

```
#include <mpshade.h>
```

```
void mpshade_setslice(size_t ntrace);
```

DESCRIPTION

This is an MPShade library combined-trace layer analyzer function.

Set the number of trace records in a "time-slice" to the given value. Future calls to **mpshade_stepid(3sh)**, **mpshade_runid(3sh)**, **mpshade_step(3sh)**, and **mpshade_run(3sh)** are affected by this call.

ERRORS

None.

SEE ALSO

mpshade_run(3sh), mpshade_step(3sh).

NAME

mpshade_stepid, mpshade_step – Return the next MPShade trace record

SYNOPSIS

```
#include <mpshade.h>
```

```
shade_trace_t *mpshade_stepid(mpshade_id_t *ptracerid);
```

```
shade_trace_t *mpshade_step(void);
```

DESCRIPTION

This is an MPShade library combined-trace layer analyzer function.

Return the next available trace record from the set of tracers managed by this analyzer.

mpshade_stepid() sets **ptracerid*

r(Retueorf

NAME

mpshade_runid, mpshade_run – Return an array of MPShade trace records

SYNOPSIS

```
#include <mpshade.h>
```

```
size_t mpshade_runid(shade_trace_t *ptrace, size_t sztrace, mpshade_id_t *ptracerid);
```

```
size_t mpshade_run(shade_trace_t *ptrace, size_t sztrace);
```

DESCRIPTION

This is an MPShade library combined-trace layer analyzer function.

Both functions write an array of trace records into the given buffer. The size of the trace buffer (in bytes) is specified by *sztrace*. For **mpshade_runid()**, all returned records are from the same tracer, whose ID is written to **ptracerid*. Unless the tracer has arranged to identify itself in the trace records, there is no way to tell which tracer generated the records returned by **mpshade_run()**.

Note, *sztrace* is the size of the buffer, not the number of trace records in the buffer. This is intentionally different from the **shade_run(3sh)** call because MPShade Trace records may have 0.0e [(Tj /F3 1 Tf 0.5 0 TD -0.092 T

MPSHade Library

mpshade_load(3sh)

NAME

mpshade

NAME

NAME

mpshadetset(3ση)

NAME

mpshade_trctl – Enable tracing in MPSHade

SYNOPSIS

```
#include <mpshade.h>
```

```
int mpshade_trctl(mpshade_iset_t *piset, shade_tri_t tri, mpshade_
```

NAME

mpshade_setmode – Switch analyzer between suspended and tracing modes

SYNOPSIS

```
#include <mpshade.h>
```

```
int mpshade_setmode(mpshade_mode_t mode);
```

DESCRIPTION

This is an MPShade library standard-tracer layer analyzer function.

If *mode* is **MPSHADE_SUSPENDED**, **mpshade_setmode()** puts the analyzer into the suspended mode.

If *mode* is **MPSHADE_TRACING**, it puts the analyzer into the tracing mode. While in suspended mode, the tracers do not generate trace data and the analyzer may change the tracing parameters. While

NAME

mpshade_sparcv9_trsz - SPARC V9 specific MPSHade trace record formats

SYNOPSIS

```
#include <mpshade_sparcv9.h>
```

