libspe Reference Manual

V1.0

Generated by Doxygen 1.4.1

Fri Sep 30 17:03:24 2005

Contents

Chapter 1

SPE Managemenent Library (libspe)

Author:
D.Herrendoerfer (d.herrendoerfer@de.ibm.com)

1.1 About

The SPE Management Library consists of PPE interfaces used to manage SPEs. These interfaces are similar to those used to manage regular threads and processes in a POSIX compliant operating system.

Library name:

libspe

Header file:

<spe.h>

<mfc.h>

1.2 Misc

This library presents work-in-progress. Some functions are provided for completeness of the programming model only.

Chapter 2

libspe Module Index

2.1 libspe Modules

Here is a list of all modules:

Chapter 3

libspe Module Documentation

3.1 Interface

Functions

- spe_program_handle_t * spe_open_image (const char *filename)

  spe_open_image - maps spe executable int system memory.

- int spe_close_image (spe_program_handle_t *program_handle)

  spe_close_image - unmaps spe executable from system memory.

- spe_gid_t spe_create_group (int policy, int priority, int use_events)

  spe_create_group - allocates a new spe thread group.

- int spe_group_max (spe_gid_t spe_gid)

  spe_group_max - returns maximum number of spe threads in this group

- speid_t spe_create_thread (spe_gid_t gid, spe_program_handle_t *handle, void *argp, void *envp, unsigned long mask, int flags)

  spe_create_thread - creates a spe thread

- int spe_wait (speid_t speid, int *status, int options)

  spe_wait - waits for spe thread to end

- int spe_kill (speid_t speid, int sig)

  spe_kill - sends a signal to a spe thread

- void * spe_get_ls (speid_t speid)

  spe_get_ls - returns the address of the local store in system memory

- int spe_get_priority (spe_gid_t gid)

  spe_get_priority - returns scheduling priority for given thread group

- int spe_set_priority (spe_gid_t gid, int priority)

  spe_set_priority - sets scheduling priority for given thread group

- int spe_get_policy (spe_gid_t gid)

  spe_get_policy - returns scheduling policy for given thread group

- spe_gid_t spe_get_group (speid_t speid)

  spe_get_group - returns group store structure a spe thread belongs to

- void * spe_get_ps (speid_t speid)

  spe_get_ps - returns pointer to problem state for mfc wrapper calls (RFU)

- int spe_group_defaults (int policy, int priority, int spe_events)

  spe_group_defaults - sets default thread group attributes.

- int _put_mfc_outbox (void *spe_ps_addr, unsigned int data)

  _put_mfc_outbox - MFC wrapper function, documented in mfc.h.

- int _stat_mfc_outbox (void *spe_ps_addr)

  _stat_mfc_outbox - MFC wrapper function, documented in mfc.h.

- int _get_mfc_inbox (void *spe_ps_addr)

  _get_mfc_inbox - MFC wrapper function, documented in mfc.h.

- int _stat_mfc_inbox (void *spe_ps_addr)

  _stat_mfc_inbox - MFC wrapper function, documented in mfc.h.

- int _stat_mfc_inbox_intr (void *spe_ps_addr)

  _stat_mfc_inbox_intr - MFC wrapper function, documented in mfc.h.

- int _send_mfc_sig (void *spe_ps_addr, unsigned int signal_reg, unsigned int data)

  _send_mfc_sig - MFC wrapper function, documented in mfc.h.

3.1.1 Function Documentation

3.1.1.1 int spe_close_image (spe_program_handle_t * *program_handle*)

spe_close_image - unmaps spe executable from system memory.

Removes memory mapped file from system memory and releases allocated storage.

Parameters:

program_handle: hanldle of the spe executable

Return values:

zero on success

-1 on failure, and sets `errno`.

3.1.1.2 spe_gid_t spe_create_group (int *policy*, int *priority*, int *use_events*)

spe_create_group - allocates a new spe thread group.

Creates a spe thread group with given attributes.

Parameters:

policy scheduling class for spe threads

priority spe groups scheduling priority

use_events RFU

Returns:

group store structure

Return values:

NULL on failure, and sets `errno`.

3.1.1.3 speid_t spe_create_thread (spe_gid_t *gid*, spe_program_handle_t * *handle*, void * *argp*, void * *envp*, unsigned long *mask*, int *flags*)

spe_create_thread - creates a spe thread

creates a spe thread of control that can be executed separately of the calling task.

Parameters:

gid spe_gid group store structure

handle a handle to a spe executable

argp ppe pointer to application specific data

envp ppe pointer to environment specific data

mask RFU

flags modifiers that are applied when the new thread is started

Returns:

spe thread store structure

Return values:

NULL on failure, and sets `errno`

3.1.1.4 spe_gid_t spe_get_group (speid_t *speid*)

spe_get_group - returns group store structure a spe thread belongs to

Parameters:

speid spe store structure

Returns:

group store structure the thread belongs to

3.1.1.5 void* spe_get_ls (speid_t *speid*)

spe_get_ls - returns the address of the local store in system memory

spe_get_ls returns the address of the spe local store in system memory.

Parameters:
speid spe store structure

Returns:
address of local store.

Return values:
NULL on failure.

3.1.1.6 int spe_get_policy (spe_gid_t *gid*)

spe_get_policy - returns scheduling policy for given thread group

Parameters:
gid spe store structure

Returns:
scheduling policy

3.1.1.7 int spe_get_priority (spe_gid_t *gid*)

spe_get_priority - returns scheduling priority for given thread group

Parameters:
gid spe group store structure

Returns:
scheduling priority.

3.1.1.8 void* spe_get_ps (speid_t *speid*)

spe_get_ps - returns pointer to problem state for mfc wrapper calls (RFU)

spe_get_ps returns a pointer to the required input for _mfc calls. In this implementation it passes the speid back.

Parameters:
speid spe store structure

Returns:
thread store structure.

3.1.1.9 int spe_group_defaults (int *policy*, int *priority*, int *spe_events*)

spe_group_defaults - sets default thread group attributes.

Parameters:

policy scheduling class for spe threads

priority spe groups scheduling priority

spe_events RFU

Return values:

zero on success

-1 on failure.

3.1.1.10 int spe_group_max (spe_gid_t *spe_gid*)

spe_group_max - returns maximum number of spe threads in this group

Parameters:

spe_gid spe_gid group store structure

Returns:

maximum number of threads in this group.

3.1.1.11 int spe_kill (speid_t *speid*, int *sig*)

spe_kill - sends a signal to a spe thread

spe_kill sends a signal to the specified process.

Parameters:

speid spe store structure

sig signal to send

Return values:

zero on success

-1 on failure.

3.1.1.12 spe_program_handle_t* spe_open_image (const char * *filename*)

spe_open_image - maps spe executable int system memory.

maps a spe executable into system memory and returns a spe_program_handle for use with spe_create_thread.

Parameters:

filename file name of the spe executable to map

Returns:

handle of the program

Return values:

NULL failure, see `errno` for more info

3.1.1.13 int spe_set_priority (spe_gid_t *gid*, int *priority*)

spe_set_priority - sets scheduling priority for given thread group

Parameters:

gid spe group store structure

priority priority to set.

Return values:

zero on success

-1 on error.

3.1.1.14 int spe_wait (speid_t *speid*, int * *status*, int *options*)

spe_wait - waits for spe thread to end

spe_wait suspende execution of the current process until the specified spe thread has exited.

Parameters:

speid spe store structure

status the spe threads return value is stored here

options RFU

Returns:

thread exit code.

Index