spu_run

Name

spu_run - execute an spu context

Synopsis

```
#include <sys/spu.h>

int spu_run(int fd, unsigned int *npc, unsigned int *event);
```

Description

The spu_run system call is used on PowerPC machines that implement the Cell Broadband Engine Architecture in order to access Synergistic Pro- cessor Units (SPUs). It uses the fd that was returned from spu_cre- ate(2) to address a specific SPU context. When the context gets sched- uled to a physical SPU, it starts execution at the instruction pointer passed in npc.

Execution of SPU code happens synchronously, meaning that spu_run does not return while the SPU is still running. If there is a need to exe- cute SPU code in parallel with other code on either the main CPU or other SPUs, you need to create a new thread of execution first, e.g. using the pthread_create(3) call.

When spu_run returns, the current value of the SPU instruction pointer is written back to npc, so you can call spu_run again without updating the pointers.

event can be a NULL pointer or point to an extended status code that gets filled when spu_run returns. It can be one of the following con- stants:

SPE_EVENT_DMA_ALIGNMENT
: A DMA alignment error

SPE_EVENT_SPE_DATA_SEGMENT
: A DMA segmentation error

SPE_EVENT_SPE_DATA_STORAGE
: A DMA storage error

If NULL is passed as the event argument, these errors will result in a signal delivered to the calling process.

Return Value

spu_run returns the value of the spu_status register or -1 to indicate an error and set errno to one of the error codes listed below. The spu_status register value contains a bit mask of status codes and optionally a 14 bit code returned from the stop-and-signal instruction on the SPU. The bit masks for the status codes are:

0x02
: SPU was stopped by stop-and-signal.

0x04
: SPU was stopped by halt.

0x08
: SPU is waiting for a channel.

0x10
: SPU is in single-step mode.

0x20
: SPU has tried to execute an invalid instruction.

0x40
: SPU has tried to access an invalid channel.

0x3fff0000
: The bits masked with this value contain the code returned from stop-and-signal.

There are always one or more of the lower eight bits set or an error code is returned from spu_run.

Errors

EAGAIN or EWOULDBLOCK
: fd is in non-blocking mode and spu_run would block.

EBADF fd is not a valid file descriptor.

EFAULT npc is not a valid pointer or status is neither NULL nor a valid pointer.

EINTR A signal occurred while spu_run was in progress. The npc value has been updated to the new program counter value if necessary.

EINVAL fd is not a file descriptor returned from spu_create(2).

ENOMEM Insufficient memory was available to handle a page fault resulting from an MFC direct memory access.

ENOSYS the functionality is not provided by the current system, because either the hardware does not provide SPUs or the spufs module is not loaded.

Notes

spu_run is meant to be used from libraries that implement a more abstract interface to SPUs, not to be used from regular applications. See http://www.bsc.es/projects/deepcomputing/linuxoncell/ for the rec- ommended libraries.

Conforming to

This call is Linux specific and only implemented by the ppc64 architec- ture. Programs using this system call are not portable.

Bugs

The code does not yet fully implement all features lined out here.

Author

Arnd Bergmann <arndb@de.ibm.com>

See Also

capabilities(7), close(2), spu_create(2), spufs(7)