NAME

pmap_enter - insert a virtual page into a physical map

SYNOPSIS

#include <sys/param.h>
#include <vm/vm.h>
#include <vm/pmap.h>

int
pmap_enter(*pmap_t pmap, vm_offset_t va, vm_page_t m, vm_prot_t prot, u_int flags, int8_t psind*);

DESCRIPTION

The **pmap_enter**() function creates a mapping in the physical map *pmap* from the virtual address *va* to the physical page *m* with the protection *prot*. Any previous mapping at the virtual address *va* is destroyed.

The *flags* argument may have the following values:

VM_PROT_READ	A read access to the given virtual address triggered the call.
VM_PROT_WRITE	A write access to the given virtual address triggered the call.
VM_PROT_EXECUTE	An execute access to the given virtual address triggered the call.
PMAP_ENTER_WIRED	The mapping should be marked as wired.
PMAP_ENTER_NOSLEEP	This function may not sleep during creation of the mapping. If the mapping cannot be created without sleeping, an appropriate Mach VM error is returned.

If the PMAP_ENTER_NOSLEEP flag is not specified, this function must create the requested mapping before returning. It may not fail. In order to create the requested mapping, this function may destroy any non-wired mapping in any pmap.

The *psind* parameter specifies the page size that should be used by the mapping. The supported page sizes are described by the global array pagesizes[]. The desired page size is specified by passing the index of the array element that equals the desired page size.

When the **pmap_enter**() function destroys or updates a managed mapping, including an existing mapping at virtual address *va*, it updates the *vm_page* structure corresponding to the previously mapped physical page. If the physical page was accessed through the managed mapping, then the *vm_page* structure's

PGA_REFERENCED aflag is set. If the physical page was modified through the managed mapping, then the **vm_page_dirty**() function is called on the *vm_page* structure.

The PGA_WRITEABLE aflag must be set for the page *m* if the new mapping is managed and writeable. It is advised to clear PGA_WRITEABLE for destroyed mappings if the implementation can ensure that no other writeable managed mappings for the previously mapped pages exist.

If the request modifies an existing mapping to use a different physical page, an implementation of **pmap_enter** must invalidate the previous mapping before installing the new one. This ensures that all threads sharing the pmap keep a consistent view of the mapping, which is necessary for the correct handling of CoW (copy on write) faults.

If the page *m* is managed, the page must be busied by the caller or the owning object must be locked. In the later case, the PMAP_ENTER_NOSLEEP must be specified by the caller.

The **pmap_enter**() function must handle the multiprocessor TLB consistency for the given address.

NOTES

On arm and i386 architectures the existing implementation of the **pmap_enter** function is incomplete, only value 0 for *psind* is supported. Other supported architectures, except amd64, have pagesizes[] array of size 1.

RETURN VALUES

If successful, the **pmap_enter**() function returns KERN_SUCCESS. If the PMAP_ENTER_NOSLEEP flag was specified and the resources required for the mapping cannot be acquired without sleeping, KERN_RESOURCE_SHORTAGE is returned.

SEE ALSO

pmap(9)

AUTHORS

This manual page was first written by Bruce M Simpson <*bms@spc.org*> and then rewritten by
Alan Cox <*alc@FreeBSD.org*> and
Konstantin Belousov <*kib@FreeBSD.org*>.