NAME

vm_map_find - find a free region within a map, and optionally map a vm_object

SYNOPSIS

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

int
vm_map_find(*vm_map_t map, vm_object_t object, vm_ooffset_t offset, vm_offset_t *addr,*
vm_size_t length, vm_offset_t max_addr, int find_space, vm_prot_t prot, vm_prot_t max, int cow);

DESCRIPTION

The **vm_map_find**() function attempts to find a free region in the target *map*, with the given *length*. If a free region is found, **vm_map_find**() creates a mapping of *object* via a call to vm_map_insert(9).

The arguments *offset*, *prot*, *max*, and *cow* are passed unchanged to vm_map_insert(9) when creating the mapping, if and only if a free region is found.

If *object* is non-NULL, the reference count on the object must be incremented by the caller before calling this function to account for the new entry.

If *max_addr* is non-zero, it specifies an upper bound on the mapping. The mapping will only succeed if a free region can be found that resides entirely below *max_addr*.

The *find_space* argument specifies the strategy to use when searching for a free region of the requested length. For all values other than VMFS_NO_SPACE, vm_map_findspace(9) is called to locate a free region of the requested length with a starting address at or above **addr*. The following strategies are supported:

VMFS_NO_SPACE	The mapping will only succeed if there is a free region of the requested length at the given address **addr*.
VMFS_ANY_SPACE	The mapping will succeed as long as there is a free region.
VMFS_SUPER_SPACE	The mapping will succeed as long as there is a free region that begins on a superpage boundary. If *object* is non-NULL and is already backed by superpages, then the mapping will require a free region that aligns relative to the existing superpages rather than one beginning on a superpage boundary.

VMFS_OPTIMAL_SPACE	The mapping will succeed as long as there is a free region. However, if *object* is non-NULL and is already backed by superpages, this strategy will attempt to find a free region aligned relative to the existing superpages.
VMFS_ALIGNED_SPACE(*n*)	The mapping will succeed as long as there is a free region that aligns on a 2^*n* boundary.

IMPLEMENTATION NOTES

This function acquires a lock on *map* by calling vm_map_lock(9), and holds it until the function returns.

The search for a free region is defined to be first-fit, from the address *addr* onwards.

RETURN VALUES

The **vm_map_find**() function returns KERN_SUCCESS if the mapping was successfully created. If space could not be found or *find_space* was VMFS_NO_SPACE and the given address, *addr*, was already mapped, KERN_NO_SPACE will be returned. If the discovered range turned out to be bogus, KERN_INVALID_ADDRESS will be returned.

SEE ALSO

vm_map(9), vm_map_findspace(9), vm_map_insert(9), vm_map_lock(9)

AUTHORS

This manual page was written by Bruce M Simpson <*bms@spc.org*>.