

**NAME**

**vm\_map\_wire**, **vm\_map\_unwire** - manage page wiring within a virtual memory map

**SYNOPSIS**

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

*int*

```
vm_map_wire(vm_map_t map, vm_offset_t start, vm_offset_t end, int flags);
```

*int*

```
vm_map_unwire(vm_map_t map, vm_offset_t start, vm_offset_t end, int flags);
```

**DESCRIPTION**

The **vm\_map\_wire()** function is responsible for wiring pages in the range between *start* and *end* within the map *map*. Wired pages are locked into physical memory, and may not be paged out as long as their wire count remains above zero.

The **vm\_map\_unwire()** function performs the corresponding unwire operation.

The *flags* argument is a bit mask, consisting of the following flags:

If the VM\_MAP\_WIRE\_USER flag is set, the function operates within user address space.

If the VM\_MAP\_WIRE\_HOLESOK flag is set, it may operate upon an arbitrary range within the address space of *map*.

If a contiguous range is desired, callers should explicitly express their intent by specifying the VM\_MAP\_WIRE\_NOHOLES flag.

**IMPLEMENTATION NOTES**

Both functions will attempt to acquire a lock on the map using **vm\_map\_lock(9)** and hold it for the duration of the call. If they detect MAP\_ENTRY\_IN\_TRANSITION, they will call **vm\_map\_unlock\_and\_wait(9)** until the map becomes available again.

The map could have changed during this window as it was held by another consumer, therefore consumers of this interface should check for this condition using the return values below.

**RETURN VALUES**

The **vm\_map\_wire()** and **vm\_map\_unwire()** functions have identical return values. The functions return **KERN\_SUCCESS** if all pages within the range were [un]wired successfully.

Otherwise, if the specified range was not valid, or if the map changed while the **MAP\_ENTRY\_IN\_TRANSITION** flag was set, **KERN\_INVALID\_ADDRESS** is returned.

### SEE ALSO

mlockall(2), munlockall(2), vm\_map(9)

### AUTHORS

This manual page was written by Bruce M Simpson <[bms@spc.org](mailto:bms@spc.org)>.