

## NAME

**vm\_map\_stack**, **vm\_map\_growstack** - manage process stacks

## SYNOPSIS

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

```
#include <vm/vm.h>
```

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

*int*

```
vm_map_stack(vm_map_t map, vm_offset_t addrbos, vm_size_t max_ssize, vm_prot_t prot,  
             vm_prot_t max, int cow);
```

*int*

```
vm_map_growstack(struct proc *p, vm_offset_t addr);
```

## DESCRIPTION

The **vm\_map\_stack()** function maps a process stack for a new process image. The stack is mapped *addrbos* in *map*, with a maximum size of *max\_ssize*. Copy-on-write flags passed in *cow* are also applied to the new mapping. Protection bits are supplied by *prot* and *max*.

It is typically called by `execve(2)`.

The **vm\_map\_growstack()** function is responsible for growing a stack for the process *p* to the desired address *addr*, similar to the legacy `sbrk(2)` call.

## IMPLEMENTATION NOTES

The **vm\_map\_stack()** function calls `vm_map_insert(9)` to create its mappings.

The **vm\_map\_stack()** and **vm\_map\_growstack()** functions acquire the process lock on *p* for the duration of the call.

## RETURN VALUES

The **vm\_map\_stack()** function returns `KERN_SUCCESS` if the mapping was allocated successfully.

Otherwise, if mapping the stack would exceed the process's VMEM resource limit, or if the specified bottom-of-stack address is out of range for the map, or if there is already a mapping at the address which would result, or if *max\_ssize* could not be accommodated within the current mapping, `KERN_NO_SPACE` is returned.

Other possible return values for this function are documented in `vm_map_insert(9)`.

The **vm\_map\_growstack()** function returns `KERN_SUCCESS` if *addr* is already mapped, or if the stack was grown successfully.

It also returns `KERN_SUCCESS` if *addr* is outside the stack range; this is done in order to preserve compatibility with the deprecated **grow()** function previously located in the file *vm\_machdep.c*.

## SEE ALSO

vm\_map(9), vm\_map\_insert(9)

## AUTHORS

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