NAME

sys/mman.h - memory management declarations

SYNOPSIS

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

DESCRIPTION

The *<sys/mman.h>* header shall define the following symbolic constants for use as protection options:

PROT_EXEC
: Page can be executed.

PROT_NONE
: Page cannot be accessed.

PROT_READ
: Page can be read.

PROT_WRITE
: Page can be written.

The *<sys/mman.h>* header shall define the following symbolic constants for use as flag options:

MAP_FIXED
: Interpret *addr* exactly.

MAP_PRIVATE
: Changes are private.

MAP_SHARED
: Share changes.

[XSI|SIO] ⌦ The *<sys/mman.h>* header shall define the following symbolic constants for the *msync()* function:

MS_ASYNC
: Perform asynchronous writes.

MS_INVALIDATE
: Invalidate mappings.

MS_SYNC
: Perform synchronous writes.

⌫

[ML] ⌦ The *<sys/mman.h>* header shall define the following symbolic constants for the *mlockall()* function:

MCL_CURRENT
: Lock currently mapped pages.

MCL_FUTURE
: Lock pages that become mapped.

⌫

The *<sys/mman.h>* header shall define the symbolic constant MAP_FAILED which shall have type **void *** and shall be used to indicate a failure from the *mmap()* function .

[ADV] ⊠ If the Advisory Information option is supported, the *<sys/mman.h>* header shall define symbolic constants for the *advice* argument to the *posix_madvise()* function as follows:

POSIX_MADV_DONTNEED
: The application expects that it will not access the specified range in the near future.

POSIX_MADV_NORMAL
: The application has no advice to give on its behavior with respect to the specified range. It is the default characteristic if no advice is given for a range of memory.

POSIX_MADV_RANDOM
: The application expects to access the specified range in a random order.

POSIX_MADV_SEQUENTIAL
: The application expects to access the specified range sequentially from lower addresses to higher addresses.

POSIX_MADV_WILLNEED
: The application expects to access the specified range in the near future.

⊠

[TYM] ⊠ The *<sys/mman.h>* header shall define the following symbolic constants for use as flags for the *posix_typed_mem_open()* function:

POSIX_TYPED_MEM_ALLOCATE
: Allocate on *mmap()*.

POSIX_TYPED_MEM_ALLOCATE_CONTIG
: Allocate contiguously on *mmap()*.

POSIX_TYPED_MEM_MAP_ALLOCATABLE
: Map on *mmap()*, without affecting allocatability.

⊠

The *<sys/mman.h>* header shall define the **mode_t**, **off_t**, and **size_t** types as described in *<sys/types.h>*.

[TYM] ⊠ The *<sys/mman.h>* header shall define the **posix_typed_mem_info** structure, which shall include at least the following member:

```
size_t  posix_tmi_length  Maximum length which may be allocated
                          from a typed memory object.
```

⊠

The following shall be declared as functions and may also be defined as macros. Function prototypes shall be provided.

[MLR]⊠

```
int    mlock(const void *, size_t);
```

⊠

[ML]⊠

```
int    mlockall(int);
```

⊠

```
void  *mmap(void *, size_t, int, int, int, off_t);
int    mprotect(void *, size_t, int);
```

[XSI|SIO]⊠

```
int    msync(void *, size_t, int);
```

⊠

[MLR]⊠

```
int    munlock(const void *, size_t);
```

⊠

[ML]⊠

```
int    munlockall(void);
```

⊠

```
int    munmap(void *, size_t);
[ADV]
int    posix_madvise(void *, size_t, int);

[TYM]
int    posix_mem_offset(const void *restrict, size_t, off_t *restrict,
           size_t *restrict, int *restrict);
int    posix_typed_mem_get_info(int, struct posix_typed_mem_info *);
int    posix_typed_mem_open(const char *, int, int);

[SHM]
int    shm_open(const char *, int, mode_t);
int    shm_unlink(const char *);
```

The following sections are informative.

APPLICATION USAGE

None.

RATIONALE

None.

FUTURE DIRECTIONS

None.

SEE ALSO

<sys/types.h>

XSH *mlock*, *mlockall*, *mmap*, *mprotect*, *msync*, *munmap*, *posix_madvise*, *posix_mem_offset*, *posix_typed_mem_get_info*, *posix_typed_mem_open*, *shm_open*, *shm_unlink*

CHANGE HISTORY

First released in Issue 4, Version 2.

Issue 5

Updated for alignment with the POSIX Realtime Extension.

Issue 6

The *<sys/mman.h>* header is marked as dependent on support for either the Memory Mapped Files, Process Memory Locking, or Shared Memory Objects options.

The following changes are made for alignment with IEEE Std 1003.1j-2000:

- The TYM margin code is added to the list of margin codes for the *<sys/mman.h>* header line, as well as for other lines.

- The POSIX_TYPED_MEM_ALLOCATE, POSIX_TYPED_MEM_ALLOCATE_CONTIG, and POSIX_TYPED_MEM_MAP_ALLOCATABLE flags are added.
- The **posix_tmi_length** structure is added.
- The *posix_mem_offset*(), *posix_typed_mem_get_info*(), and *posix_typed_mem_open*() functions are added.

The **restrict** keyword is added to the prototype for *posix_mem_offset*().

IEEE PASC Interpretation 1003.1 #102 is applied, adding the prototype for *posix_madvise*().

IEEE Std 1003.1-2001/Cor 1-2002, item XBD/TC1/D6/16 is applied, correcting margin code and shading errors for the *mlock*() and *munlock*() functions.

IEEE Std 1003.1-2001/Cor 1-2002, item XSH/TC1/D6/34 is applied, changing the margin code for the *mmap*() function from MF|SHM to MC3 (notation for MF|SHM|TYM).

IEEE Std 1003.1-2001/Cor 1-2002, item XSH/TC1/D6/36 is applied, changing the margin code for the *munmap*() function from MF|SHM to MC3 (notation for MF|SHM|TYM).

Issue 7

SD5-XBD-ERN-5 is applied, rewriting the DESCRIPTION.

Functionality relating to the Memory Protection and Memory Mapped Files options is moved to the Base.

This reference page is clarified with respect to macros and symbolic constants.

End of informative text.

return to top of page

[Main Index | XBD | XSH | XCU | XRAT]

<<< Previous Home Next >>>