



*Full credit is given to the above companies including the OS that this PDF file was generated!*

## **Red Hat Enterprise Linux Release 9.2 Manual Pages on 'posix\_mem\_offset.3p' command**

**\$ man posix\_mem\_offset.3p**

POSIX\_MEM\_OFFSET(3P) POSIX Programmer's Manual POSIX\_MEM\_OFFSET(3P)

### PROLOG

This manual page is part of the POSIX Programmer's Manual. The Linux implementation of this interface may differ (consult the corresponding Linux manual page for details of Linux behavior), or the interface may not be implemented on Linux.

### NAME

posix\_mem\_offset ? find offset and length of a mapped typed memory block (ADVANCED REALTIME)

### SYNOPSIS

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

int posix_mem_offset(const void *restrict addr, size_t len,
    off_t *restrict off, size_t *restrict contig_len,
    int *restrict fildes);
```

### DESCRIPTION

The `posix_mem_offset()` function shall return in the variable pointed to by `off` a value that identifies the offset (or location), within a memory object, of the memory block currently mapped at `addr`. The function shall return in the variable pointed to by `fildes`, the descriptor used (via `mmap()`) to establish the mapping which contains `addr`. If that descriptor was closed since the mapping was established, the returned value of `fildes` shall be `-1`. The `len` argument specifies the length of the block of the memory object the user wishes the offset for; upon re-

turn, the value pointed to by `contig_len` shall equal either `len`, or the length of the largest contiguous block of the memory object that is currently mapped to the calling process starting at `addr`, whichever is smaller.

If the memory object mapped at `addr` is a typed memory object, then if the `off` and `contig_len` values obtained by calling `posix_mem_offset()` are used in a call to `mmap()` with a file descriptor that refers to the same memory pool as `fildev` (either through the same port or through a different port), and that was opened with neither the `POSIX_TYPED_MEM_ALLOCATE` nor the `POSIX_TYPED_MEM_ALLOCATE_CONTIG` flag, the typed memory area that is mapped shall be exactly the same area that was mapped at `addr` in the address space of the process that called `posix_mem_offset()`.

If the memory object specified by `fildev` is not a typed memory object, then the behavior of this function is implementation-defined.

## RETURN VALUE

Upon successful completion, the `posix_mem_offset()` function shall return zero; otherwise, the corresponding error status value shall be returned.

## ERRORS

The `posix_mem_offset()` function shall fail if:

**EACCES** The process has not mapped a memory object supported by this function at the given address `addr`.

This function shall not return an error code of `[EINTR]`.

The following sections are informative.

## EXAMPLES

None.

## APPLICATION USAGE

None.

## RATIONALE

None.

## FUTURE DIRECTIONS

None.

## SEE ALSO

`mmap()`, `posix_typed_mem_open()`

The Base Definitions volume of POSIX.1-2017, `<sys_mman.h>`

## COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1-2017, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 7, 2018 Edition, Copyright (C) 2018 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.opengroup.org/unix/online.html>.

Any typographical or formatting errors that appear in this page are most likely to have been introduced during the conversion of the source files to man page format. To report such errors, see [https://www.kernel.org/doc/man-pages/reporting\\_bugs.html](https://www.kernel.org/doc/man-pages/reporting_bugs.html).

IEEE/The Open Group

2017

POSIX\_MEM\_OFFSET(3P)