

IEEE Standards Interpretations for IEEE Std 1003.1b™-1993 IEEE Standard for Information Technology - Portable Operating System Interfaces (POSIX(R)) - Part 1: System Application Program Interface (API) - Amendment 1: Realtime Extension [C language]

Copyright © 1996 by the Institute of Electrical and Electronics Engineers, Inc. 3 Park Avenue New York, New York 10016-5997 USA All Rights Reserved.

Interpretations are issued to explain and clarify the intent of a standard and **do not** constitute an alteration to the original standard. In addition, interpretations are not intended to supply consulting information. Permission is hereby granted to download and print one copy of this document. Individuals seeking permission to reproduce and/or distribute this document in its entirety or portions of this document must contact the IEEE Standards Department for the appropriate license. Use of the information contained in this document is at your own risk.

IEEE Standards Department Copyrights and Permissions 445 Hoes Lane, Piscataway, New Jersey 08855-1331, USA

Interpretation Request #3

Topic: mmap **Relevant Sections:** 12.2.1.2 **Classification:** (to be assigned)

I'd like to receive clarification on text I read in 1003.1b-1993. Specifically the question has to do with mmap() and what is meant by the following (removed word for word from page 236, lines 213-215): "The mapping established by mmap() replaces any previous mappings for those whole pages containing any part of the process's address space starting at "pa" and continuing for "len" bytes."

There are a couple of ways this can be interpreted and I'd like to find out which is the intended one. a) Does this mean if I map page 3 of a file in one call and page 5 in another call and then map pages 3-5 in a separate call, the call succeeds and the previous mappings of page 3 and page 5 are no longer valid? If so, does that apply to both MAP_PRIVATE and MAP_SHARED? b) Does this apply to MAP_FIXED where lets say the user has page 1 of file A mapped at address 0x40005000 and then a mmap (MAP_FIXED) at address 0x40004000 for three pages (range 0x40004000-0x40006fff) on file B overwrites the other mmap? In other words am I allowed overlay mappings with different files? If so, both MAP_SHARED and MAP_FIXED? c) Does the overwrite only apply to mmap() collisions or any address collision. For instance, can a process mmap() over its data object or other non mmap objects?

Interpretation Response

For question a), the standard is clear that the mapping is replaced the addresses overlap but not replaced where they don't. Page 238 line 274: "...nor shall it replace an extant mapping." Part one of the question: ... the other mappings are no longer valid? - this is incorrect, unless the addresses are specified in MAP_FIXED and overlap. The standard is

clear that this applies to both MAP_PRIVATE and MAP_SHARED.

Question b, the standard is clear that you can replace the existing mappings using MAP_FIXED independent of MAP_SHARED and MAP_PRIVATE. Question c, the standard is silent on whether you can map over objects other than ones mapped by MMAP. Implementations can allow MAP_FIXED mapping over other kinds of memory or not allow it.

Rationale for Interpretation

None.