

IEEE Standards Interpretations for IEEE Std 1003.1c™-1995 IEEE Standard for Information Technology--Portable Operating System Interface (POSIX(R)) - System Application Program Interface (API) Amendment 2: Threads 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 #29

Topic: Get scheduling parameter limits **Relevant Clauses:** 13.3.6

The standard fails to formalize the dependency of `_POSIX_THREAD_PRIORITY_SCHEDULING` on `_POSIX_PRIORITY_SCHEDULING`. That is, a conforming implementation could choose to support the latter but not the former. That was intentional, BUT, there really is a “hidden” dependency... to use thread priority scheduling one must call `sched_priority_get_min()` and/or `sched_priority_get_max()` to determine the legal priority range for any policy. And those functions are required only under `_POSIX_PRIORITY_SCHEDULING`, and therefore might not exist.

I would suggest one of three solutions:

- 1) require `_POSIX_PRIORITY_SCHEDULING` if the implementation supports `_POSIX_THREAD_PRIORITY_SCHEDULING`, or,
- 2) `sched_priority_get_min/_max` must be required if EITHER option is supported, or,
- 3) go back to the wording in one of the earlier drafts, where there was no separate thread priority scheduling option at all -- the thread scheduling functions were present IFF the system supported threads and priority scheduling.

Option 2 is likely to be the least contentious of the three, and it's certainly sufficient.

Interpretation Response

The standard is clear in 13.2 that priorities shall be in the range defined for the policy and in section 13.3.6 it only requires the `sched_get_priority_max` and `sched_get_priority_min` function if `_POSIX_PRIORITY_SCHEDULING` is defined. The standard's con-

formance requirement upon these functions when that option is not defined is only that if the functions are provided, they shall be provided as specified or they shall fail. The standard is also clear that the `_POSIX_THREAD_PRIORITY_SCHEDULING` option does not require that the `sched_get_priority_max` and `sched_get_priority_min` functions be defined. The interpretations committee believes that not requiring the definition of these functions may make the `_POSIX_THREAD_PRIORITY_SCHEDULING` option less useful and was not the intention of the working and balloting groups. This is being referred to the sponsor for consideration.

Rationale for Interpretation

None.