

## IEEE Standards Interpretation for IEEE Std 1003.1™-2001 IEEE Standard Standard for Information Technology -- Portable Operating System Interface (POSIX®)

Copyright © 2006 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 #70

**Topic:** `select()` `FD_SETSIZE` **Relevant Sections:** XSH `select()` Page: 0 Line: 0

Historical kernel implementations of `select()` have used an `auto (stack)` declaration of an `fd_set` (bit array) object and copied the data from user space into this object. `Select()`, however, has evolved, removing the static limitation of `FD_SETSIZE` descriptors. Current implementations allow a value for the first parameter, `nfds`, to be in excess of `FD_SETSIZE`.

The kernel accomplishes this by internally allocating a bit array sufficient to contain the number of bits needed to support an arbitrary number of bits being selected upon. The length of the copy into the kernel allocated array is based on the value of `nfds`, rather than on the value of the manifest constant `FD_SETSIZE`.

As a result, this permits guarding of the `FD_SETSIZE` definition in `<unistd.h>`, and thus permits user programs to redefine the value prior to including the system header. The user definition of this value is then used in preference to the system default value.

A consequence of this is that it's possible to pass into `select()` a value for the first parameter - `nfds` - in excess of the system's `FD_SETSIZE` default value.

Implementing this way removes the historical limitation of `FD_SETSIZE` as a fixed maximum value for `nfds`.

This functionality was first introduced into BSD 4.4-based systems, and is utilized by many existing applications.

The standard states that `select()` “shall” fail and set `errno` to `[EINVAL]` when the `nfds` argument is less than 0 or greater than `FD_SETSIZE`.

The ERRORS section of the standard should be changed to accomodate more up-to-date implementations of `select()` so that errors for `nfds > FD_SETSIZE` “may fail” .

The wording in the ERRORS section where `FD_SETSIZE` is mentioned, should be changed to allow `nfds` arguments greater than the `FD_SETSIZE` without generating an error. So, the `EINVAL` error for `nfds > FD_SETSIZE` should change from “shall fail” to “may fail” .

In the “shall fail” list of errors, REMOVE “[EINVAL] The `nfds` argument is less than 0 or greater than `FD_SETSIZE`.”

REPLACE WITH “[EINVAL] The `nfds` argument is less than 0.”

AND ADD a “may fail” list that states: “Under the following conditions, `pselect()` and `select()` “may” fail and set `errno` to: `[EINVAL]` The `nfds` argument is greater than `FD_SETSIZE`.”

REPLACE “[EINVAL] The `nfds` argument is less than 0 or greater than `FD_SETSIZE` ”

WITH “[EINVAL] The `nfds` argument is less than 0. `[EINVAL]` The `nfds` argument is greater than the default limit for `FD_SETSIZE` and the default limit for `FD_SETSIZE` is enforced by the system.”

### **Interpretation Response #70**

The standard clearly states that `FD_SETSIZE` is a fixed limit, and conforming implementations must conform to this

### **Rationale for Interpretation**

The redefinition of `FD_SETSIZE` by applications is non-conforming behavior, and the effects are not specified in the standard. It was also felt that applications can use the `poll()` interface for handling larger numbers of file descriptors.