

## 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 #55

**Topic:** system() thread-safety, at\_fork handlers **Relevant Sections:** XSH 2.9.1, system() Page: 0 Line: 0

1. It is unclear whether calling system(3) invokes atfork handlers in a conforming implementation. system(3) specifies

"The environment of the executed command shall be as if a child process were created using fork(), and the child process invoked the sh utility using exec() as follows:"

In particular, usage of the word "environment" is confusing here. It may refer just to environment variables, however, the "Application usage" sections indicates that also signal handlers should be arranged as if the process was created through fork() and exec(). This still makes not clear whether handlers installed through pthread\_atfork() are invoked.

2. The system() function is defined to be a thread-safe function. The specification requires it to use features which have an affect at the process scope, which renders it non-thread-safe.

As per the system () interface definition, (XSH P1540 ,L46753-46754 "The system() function shall ignore the SIGINT and SIGQUIT signals, and shall block the SIGCHLD signal"), the implementation needs to ignore the SIGINT and SIGQUIT signals during the execution of system(). To achieve this the implementation needs to literally or effectively execute the sigaction() function which has process wide scope (XSH P1402, L42498,42499, the sigaction() function allows the calling process to examine and/or specify the action to be associated with a specific signal).

Since `sigaction()` can also be called in another thread to set a different signal action for `SIGINT` and `SIGQUIT`. This makes `system()` non-threadsafe.

1. The description of `system()` should change to “`system()` behaves as if a new process was created using `fork()`, and the child process invoked the `sh` utility using `execl()` ...”

In addition, the application usage section should make it clear that `atfork` handlers are invoked.

2. In System Interfaces, section 2.9.1, line 2089, add “`system()`” to the list of functions that need not be thread-safe.

### **Interpretation Response #55**

1. The standard does not speak to the issue of `at_fork()` handlers, and as such no conformance distinction can be made between alternative implementations based on this. This is being referred to the sponsor.

2. The standard states the requirements for thread-safety for `system()`, and conforming implementations must conform to this. However, concerns have been raised about this which are being referred to the sponsor.

### **Rationale for Interpretation**

None.