IEEE P1003.2 Draft 11.2 – September 1991

Annex B
(norm ative)

C Language Bindings Option

This annex describes the C language bindings to the language-independent services described in Section 7.

The interfaces described in this annex may be provided by the conforming system; however, any system claiming conformance to the Language-Independent System Services C Language Bindings Option shall provide all of the interfaces described here.

B.0.1 C Language Bindings Option Rationale. *(This subclause is not a part of P1003.2)*

In this version of POSIX.2, the language-independent descriptions in Section 7 have not been developed. The language-independent syntax is being created in parallel by the POSIX.1 working group. Therefore, the C language bindings described in this annex are actually the full functional specifications. It is the intention of the POSIX.2 working group to rectify this situation in a revision to this standard, by moving the majority of the functional specifications back into Section 7, leaving Annex B with only brief descriptions of the C bindings to those services.

B.1 C Language Definitions

B.1.1 POSIX Symbols

Certain symbols in this annex are defined in headers. Some of those headers could also define symbols other than those defined by this standard, potentially conflicting with symbols used by the application. Also, this standard defines symbols that other standards do not permit to appear in those headers without some control on the visibility of those symbols.

Symbols called *feature test macros* are used to control the visibility of symbols that might be included in a header. Implementations, future versions of this standard, and other standards may define additional feature test macros. The `#define`s for feature test macros shall appear in the application source code before any `#include` of a header where a symbol should be visible to some, but not all, applications. If the definition of the macro does not precede the `#include`, the result is undefined.

Feature test macros shall begin with the underscore character (_) and an upper- 1
case letter, or with two underscore characters. 1

Implementations may add symbols to the headers shown in Table B-1, provided 1
the identifiers for those symbols begin with the corresponding reserved prefixes in 1
Table B-1. Similarly, implementations may add symbols to the headers in 1
Table B-1 that end in the string indicated as a reserved suffix as long as the 1
reserved suffix is in that part of the name considered significant by the implemen- 1
tation. This shall be in addition to any reservations made in the C Standard {7}. 1

After the last inclusion of a given header, an application may use any of the sym- 1
bol classes reserved in Table B-1 for its own purposes, as long as the requirements 1
in the note to Table B-1 are satisfied, noting that the symbol declared in the 1
header may become inaccessible. 1

Future revisions of this standard, and other POSIX standards, are likely to use 1
symbols in these same reserved spaces. 1

In addition, implementations may add members to a structure or union without 1
controlling the visibility of those members with a feature test macro, as long as a 1
user-defined macro with the same name cannot interfere with the correct 1
interpretation of the program. 1

A conforming POSIX.2 application shall define the feature test macro in Table B-2. When an application includes a header and the _POSIX_C_SOURCE feature test macro is defined to be the value 1 or 2, the effect shall be the same as if _POSIX_SOURCE was defined as described in POSIX.1 {8}.

In addition, when the application includes any of the headers defined in this stan- 1
dard, and _POSIX_C_SOURCE is defined to be the value 2: 1

(1) All symbols defined in POSIX.2 to appear when the header is included
shall be made visible. 1