1 Introduction

(1) The ANSI/ISO C standard draws a distinction between freestanding and hosted implementations. Hosted implementations must provide the full set of standard-defined libraries. Freestanding implementations, on the other hand, have an implementation-defined set of libraries available.

(2) The purpose for this distinction is to allow conforming C implementations targeted to systems in which a full operating system is not available. For example, a conforming (freestanding) C implementation could generate code to execute in a microwave oven or in the braking system of an automobile. It would not make sense to require such an implementation to support standard I/O.

(3) I believe this distinction is valuable for the C++ as well, so that the embedded systems market can be served by a standard that applies to its processors and programs.

2 Proposal

(1) I propose to add the following to the WP, in addition to what is in X3J16/93-0063 = WG21/N0270 (voted into the draft at Munich, motion 3) section A “Processor Compliance” to distinguish between two kinds of implementations: freestanding and hosted:

3. Two kinds of implementations are defined: hosted and freestanding. For a hosted implementation, this standard defines the set of available libraries. A freestanding implementation is one in which execution may take place without the benefit of an operating system, and has an implementation-defined set of libraries. This set shall include the C++ language support libraries (Section 17.1), the C language support libraries (float.h), limits.h, stdarg.h, and stddef.h), and the functions abort(), atexit(), and exit() in stdlib.h.

(2) These changes are modelled after the “Compliance” and “Freestanding Environment” sections of the ISO C standard (ANSI X3.159-1989 sections 1.7 and 2.1.2.1), with the three functions abort, exit, and atexit added because they are explicitly referred to elsewhere in the standard.