Conformance of Freestanding Implementations

Doc No:	**X3J16/93-0046**
	WG21/N0253
	c++std-env-369
Date:	**October 21, 1993**
Project:	Programming Language C++
Reply-To:	Neal M Gafter
	gafter@mri.com

1 Introduction

(1) The ANSI/ISO C standard draws a distinction between *freestanding* and *hosted* implementations. Hosted implementations have behaviors for program start and termination that are defined by the standard, and require the full set of standard-defined libraries to be available. Freestanding implementations, on the other hand, have implementation-defined semantics for program start and termination, and 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 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 X3J16/93-0063 (WG21/N0270) 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, semantics for program start, and semantics for program termination. 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, semantics for program start, and semantics for program termination. See also section 3.4 “Start and Termination”.

(2) I further propose to modify the Working Paper section 3.4 “Start and Termination” to allow the entire section to apply to hosted implementations only, and to require a freestanding implementation to define a replacement for this section. Add something like the following to the beginning of section 3.4:

> The semantics of program start and termination, and the timing of static constructors and destructors is implementation-defined for freestanding implementations. The rest of this section applies to hosted implementations.