Document Number: WG14 N606/X3J11 96-070

C9X Revision Proposal

Title: Extended integers promotion rules
Author: Frank Farance
Author Affiliation: Farance Inc.
Postal Address: 555 Main Street, New York, NY, 10044-0150, USA
E-mail Address: frank@farance.com
Telephone Number: +1 212 486 4700
Fax Number: +1 212 759 1605
Sponsor: X3J11
Date: 1996-09-13
Proposal Category:
__ Editorial change/non-normative contribution
__ Correction
X_ New feature
__ Addition to obsolescent feature list
__ Addition to Future Directions
__ Other (please specify) _____
Area of Standard Affected:
__ Environment
X_ Language
__ Preprocessor
__ Library
__ Macro/typedef/tag name
__ Function
__ Header
Prior Art: "<inttypes.h>" and "long long" proposals.
Target Audience: C programmers
Related Documents (if any):
Proposal Attached: X_ Yes __ No, but what's your interest?

With the inclusion of the "<inttypes.h>" proposal, it is possible to provide more numeric types than C's integral types: "char", "short", "int", "long", and (recently) "long long". For example, an implementation might provide 8-bit ("char"), 16-bit ("short"), 32-bit ("int" and "long"), 48-bit (no corresponding C type), 64-bit ("long long"), and 128-bit (no corresponding C type). The "<inttypes.h>" proposal gives access to this compiler ''magic'' tokens via macros defined in "<inttypes.h>". This proposal simplifies the promotion rules while allowing as many integral types as the implementor desires.

PROPOSED WORDING CHANGES

In subclause 6.2.1.5, Usual arithmetic conversions, replace the text staring with ''If either operand has type "unsigned long int" ...'' and ending with ''Otherwise, both operands have type "int".'' with the following text:

> The signedness (footnote 1) and precision (footnote 2) is determined for both operands. Operands with

precision less than "int" are promoted to "int" if the operand is signed or to "unsigned int" if the operand is unsigned. The type of the result is the type of the operand with the larger precision. For operands with the same precision, but differing types, the C types shall be ordered from largest to smallest: signed and unsigned "long long", signed and unsigned "long", signed and unsigned "int". The result shall be unsigned if the operands differ in signedness, otherwise the result shall have signedness of the operands.

Footnote #1: The signedness is either signed or unsigned.

Footnote #2: Precision is measured in bits available for storage. For example, both "unsigned int" and "int" have the same number of bits available for storage, but their range of values differs.