C9X Complex Arithmetic Proposal—Overview

WG14/N515 X3J11/95-116 (Draft 12/21/95)

Jim Thomas
Taligent, Inc.
10201 N. DeAnza Blvd.
Cupertino, CA 95014-2233
jim_thomas@taligent.com

The three papers

Complex Arithmetic—C9X Edits (N516)
Complex Arithmetic <complex.h> (N517)
Complex Arithmetic for IEEE Implementations (N518)

collectively constitute a detailed proposal to integrate the "Complex C Extensions" chapter of X3J11's *Numerical C Technical Report* (X3/TR-17:199x) into C9X. The first two papers pertain to all C9X implementations, the last to those providing IEEE arithmetic.

Background

At its initial meeting in 1989, the Numerical C Extensions Group (NCEG), now incorporated into the ANSI C language committee (X3J11), targeted complex arithmetic as one of the areas requiring supporting extensions, in order for C to become suitable for general numerical programming.

The first NCEG complex proposal, "Complex Extension to C" (Knaak), followed a traditional FORTRAN-like approach, adding complex data types as ordered pairs of real numbers and specifying the arithmetic of these types. "Augmenting a Programming Language with Complex Arithmetic" (Kahan and Thomas) showed certain semantic and efficiency problems inherent in the traditional approach, including fundamental incompatibilities with advances in IEEE standard real arithmetic. To solve these problems, the current specification refines the traditional approach by adding both complex and imaginary types and specifying arithmetic on any combinations of real, imaginary, and complex operands. "Issues Regarding Imaginary Types for C and C++" (Thomas and Coonen) provides detailed rationale. Also, this specification is designed for consistency with the floating-point proposal for C9X.

At its June 1994 meeting, X3J11 approved following the direction of this proposal. At its December 1994 meeting, X3J11 accepted the specification for inclusion in its Technical Report. Currently, however, both this proposal and the traditional FORTRAN-like proposal are still under consideration for C9X.

Substantive changes from CCE

Specification for IEEE implementations has been added, including treatment of infinities. (The specification of special cases for library functions is still in progress.) *Rationale*: This was needed to support a programming model that takes advantage of IEEE features.

The macro scheme for the imaginary unit has been improved to not require a keyword in the implementation's name space. *Rationale*: This avoids defining a keyword.

The overloading specification has been updated to match changes in the floating-point specification. *Rationale*: This is for basic consistency, plus the advantages of the new scheme, described in the floating-point proposal.

Imaginary and complex expressions have been disallowed as arguments for prefix increment and decrement operators. *Rationale*: Allowing an imaginary or complex operand for these operators seems potentially confusing and not particularly useful.

The imaginary and complex types are now designated without an underscore character, e.g. `double complex` instead of `double_complex`. *Rationale*: This is a better fit for the C standard. The underscore approach was intended to allow C++ compatible code, which now requires more extreme measures for other reasons.

The designations `long complex` and `long imaginary` are allowed for `long double complex` and `long double imaginary`, respectively. *Rationale*: Convenience of brevity.

Acknowledgments

People who have made especially substantial contributions to this document include, in alphabetical order: Jerome Coonen, W. Kahan, Clayton Lewis, David Prosser, and Fred Tydeman.

Others who have provided invaluable contributions, reviews, or administrative help include, in alphabetical order: Frank Farance, Harry H. Cheng, Yinsun Feng, W.M. Gentleman, Bill Gibbons, Ron Guilmette, Doug Gwyn, David Hough, Rex Jaeschke, David Keaton, David Knaak, John Kwan, Tom MacDonald, Stuart McDonald, Colin McMaster, Randy Meyer, Randy Meyers, Adolfo Nemirovsky, Jon Okada, P.J. Plauger, Thomas Plum, and Pat Ricci.