Keywords _Complex and _Imaginary

WG14/N809 J11/98-008 (Draft 1998-02-05)

Jim Thomas
`jthomas@best.com`

This document states the wording changes to C9x draft N794 to replace the keywords **`complex`** and **`imaginary`** with **`_Complex`** and **`_Imaginary`**, and to exempt freestanding implementations from the requirement to support complex arithmetic.

6.1.1 Keywords, p33

In the list of keywords, replace **`complex`** and **`imaginary`** with **`_Complex`** and **`_Imaginary`**.

Replace the Semantics section with the C89 wording.

6.1.2.5 Types, p39

In Paragraphs 10 and 11 replace four instances of **`complex`** (in program font) with **`_Complex`**.

6.5.2 Type specifiers, p102-3

In [1] change **`complex`** to **`_Complex`** and add **`_Imaginary`**.

In Constraints, following the list, add:

The type specifiers **`_Complex`** and **`_Imaginary`** shall not be used if the implementation does not provide those types.*

* Implementations are not required to provide imaginary types. Freestanding implementations are not required to provide complex types.

In [2] replace three instances of **`complex`** with **`_Complex`**, and add to the list **`float _Imaginary`**, **`double _Imaginary`**, and **`long double _Imaginary`**.

Footnote 18, p26

Replace two instances of **`complex`** with **`_Complex`**.

Footnote 49, p63

Replace two instances of **`complex`** with **`_Complex`**.

B.1.3 Keywords, p436

Replace **`complex`** by **`_Complex`**, and **`imaginary`** by **`_Imaginary`**.

B.2.2 Declarations, p444

Replace **`complex`** by **`_Complex`**, and add **`_Imaginary`**.

7.8 Complex arithmetic <complex.h>, p249

Replace paragraphs 2 and 3 by:

The macro

```
complex
```

is defined to be **`_Complex`**.

The macro

```
_Complex_I
```

expands to a constant expression of type **`const float _Complex`**, with the value of the imaginary unit.[185]

The macro

```
imaginary
```

is defined if and only if the implementation supports imaginary types. [186] If defined, it is defined to be the keyword **`_Imaginary`**.

The macro

```
_Imaginary_I
```

is defined if and only if the macro **`imaginary`** is defined. If defined, it expands to a constant expression of type **`const float _Imaginary`**, with the value of the imaginary unit.

The macro

```
I
```

is defined to be either `_Imaginary_I` or `_Complex_I`. If `_Imaginary_I` is not defined, `I` shall be defined to be `_Complex_I`.

Notwithstanding the provisions of subclause 7.1.3, it is permitted to undefine and perhaps then redefined the macros `complex`, `imaginary`, and `I`. [NOTE TO EDITOR: Should “subclause” be removed from the the preceding sentence?]

G.2 Types, p508

In [1] replace three instances of `imaginary` with `_Imaginary` [NOTE TO EDITOR: program font is needed here]. In [2] replace `imaginary` with `_Imaginary`.

G.4.1 Multiplicative operators, p510, 512

In [6] in the first line of each example change `complex` to `_Complex`.

G.5 <complex.h>, p514

Replace the first paragraph with

The macros

```
imaginary
_Imaginary_I
```

are defined, and the macro

```
I
```

is defined to be `_Imaginary_I` (7.8).

K.2 Undefined behavior, p535

Remove the bullet “The reserved token `complex` or `imaginary` is used before <complex.h> is included.”