Document: N1511
Date: 2010/09/23
Author: Jim Thomas and Fred Tydeman
Subject: Clarifications for wide evaluation

Clause 6.3.1.4 speaks of converting "a finite value of real floating type". This appears to mean a value representable in the type, which does not include values represented in greater precision and range than their semantic type. The following change includes them:

6.3.1.4 #1, first sentence: change "When a finite value of real floating type is converted to an integer type …" to "When a real finite floating-point value is converted to an integer type …"

Clause 6.3.1.5 refers to promoting or demoting a floating type. This is ambiguous - maybe meaning a value representable in the type, or maybe a value with extra precision and range. Paragraph #2 has words intended to cover the case of extra precision and range, but paragraph #1 does not. A change in the style of the one above for 6.3.1.4 is clearer and covers the specification in both paragraphs:

6.3.1.5 #1, #2 change

> When a **`float`** is promoted to **`double`** or **`long double`**, or a **`double`** is promoted to **`long double`**, its value is unchanged (if the source value is represented in the precision and range of its type).
>
> When a **`double`** is demoted to **`float`**, a **`long double`** is demoted to **`double`** or **`float`**, or a value being represented in greater precision and range than required by its semantic type (see 6.3.1.8) is explicitly converted (including to its own type), if the value being converted can be represented exactly in the new type, it is unchanged. If the value being converted is in the range of values that can be represented but cannot be represented exactly, the result is either the nearest higher or nearest lower representable value, chosen in an implementation-defined manner. If the value being converted is outside the range of values that can be represented, the behavior is undefined.

to

> When a real floating-point value is converted to a floating type, if the value being converted can be represented exactly in the new type, it is unchanged. If the value being converted is in the range of values that can be represented but cannot be represented exactly, the result is either the nearest higher or nearest lower representable value, chosen in an implementation-defined manner. If the value being converted is outside the range of values that can be represented, the behavior is undefined.