Document: WG14/N1218
Date: 2007/03/26
Project: TR 24732
Authors: Jim Thomas, Rich Peterson
Reply to: Rich Peterson <Rich.Peterson@hp.com>

Subject: component extraction

This paper notes the difficulty in determining the components in the abstract representation of a decimal floating-point number, and proposes adding to WG14/N1201 (TR 24732 draft of 2006/11/10) a function for this purpose.

Background: 754R specifies decimal floating-point arithmetic in terms of an abstract representation with integer components (sign, coefficient, quantum exponent). A finite decimal value generally has multiple representations, e.g. the numerical value 1.23 is represented by (1, 123, -2), (1, 1230, -3), etc. See N1214 for details.

The frexp functions compute the normalized significand and exponent as defined in the traditional C model in 5.2.4.2.2. However, this does not suffice to determine the coefficient and quantum exponent for decimal, because the multiple representations of a decimal value all have the same normalized significand and exponent.

For generic floating types with binary radix, programmers commonly extract bits directly from the encoding and use them in a simple computation to determine the representation components. For decimal floating types, computation of the representation components from the encoding bits is much more complicated, because the information is obscured by the encoding (packing) of decimal digits into bits and because there are two encoding schemes (see N1219).

There are several ways C could provide support for component extraction for decimal. This document proposes a minimalist approach of adding a "quantexp" function, for each of the decimal floating types, that returns the quantum exponent of its argument. A quantexp function can be used with scalbn to compute the signed coefficient in decimal floating format. For example, for finite _Decimal64 x, the signed coefficient is equal to scalbnd64(x, -quantexpd64(x)), which printf with %.0DF renders as an integer decimal character sequence. Note that integer types might not have enough precision to represent coefficients.

Suggested TR changes:

In 9.4, add the following suggested addition to C99:

7.12.11.7 The quantexp functions

Synopsis

```
#define __STDC_WANT_DEC_FP__
#include <math.h>
int quantexpd64(_Decimal64 x);
int quantexpd32(_Decimal32 x);
int quantexpd128(_Decimal128 x);
```

Description

The quantexp functions compute the quantum exponent of a finite argument. If x is infinite or NaN, they compute INT_MIN and a domain error occurs.

Returns

The quantexp functions return the quantum exponent of x.