Interchange and extended types

TS 18661 Part 3
N1691
Jim Thomas  2013-04-25
Interchange formats

- IEEE 754:208 introduced a "tower" of interchange formats
- Arbitrarily large widths (32x)
- Precision and range determined by width
- binary16, for GPU data etc.
- For exchange of FP data
- May or may not be arithmetic
Extended formats

- IEEE specifies *extended* formats that extend its basic formats: binary32|64|128 and decimal64|128
- Have at least a specified precision and range
- For explicit wide evaluation
- Not for data exchange
Interchange and extended types

• TS 18661 Part 3 is C binding for new formats to new C types
• N1691 is first draft to appear in WG 14 mailing
• Believed complete
• Goal: get input, update for next meeting, have ready for WG 14 review before meeting after that
• So following along one meeting behind Part 2
• P1 – conformance requires conformance to Part 1 or Part 2 (or both)

• ISSUE 1 – interdependencies among Parts
  – Might have Part 1 + Part 3, Part 2 + Part 3, or Part 1 + Part 2 + Part 3
  – Effective spec is C11 with changes in supported Parts
Type structure additions

data-interchange types (\_FloatN, \_DecimalN)
  interchange floating types (which are arithmetic)
    decimal floating types
    non arithmetic data-interchange types

extended floating types (\_FloatNx, \_DecimalNx)

real floating types
  generic floating types
    interchange floating types
    extended floating types

basic types
  char    signed and unsigned integer    floating types
  data-interchange types

scalar types
  arithmetic types    pointer types
  data-interchange types
Type structure unchanged

floating types
  real floating types
  complex types

real types
  integer types
  real floating types

arithmetic types
  integer types
  floating types
• P2 – types are distinct and not compatible
• P4 – **requires** interchange and extended floating types whose formats must already be supported because of conformance to Part 1 or 2
• P4 – **requires** `_Float16` at least as data-interchange type (if Part 1 conformance)
• P5 – **requires** complex (and imaginary) types for supported binary interchange and extended floating types
Example 1

Assume

– Part 1 conformance
– long double has common IEEE 80-bit extended format

Types

<table>
<thead>
<tr>
<th>Width</th>
<th>Generic</th>
<th>Interchange</th>
<th>Extended</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td></td>
<td>_Float16</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>float</td>
<td>_Float32</td>
<td></td>
</tr>
<tr>
<td>64</td>
<td>double</td>
<td>_Float64</td>
<td>_Float32x</td>
</tr>
<tr>
<td>80</td>
<td>long double</td>
<td></td>
<td>_Float64x</td>
</tr>
</tbody>
</table>

_Float32x could have the 80-bit format
Example 2

Assume

– Part 1 conformance
– long double has IEEE binary128 format

Types

<table>
<thead>
<tr>
<th>Width</th>
<th>Generic</th>
<th>Interchange</th>
<th>Extended</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td></td>
<td>_Float16</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>float</td>
<td>_Float32</td>
<td></td>
</tr>
<tr>
<td>64</td>
<td>double</td>
<td>_Float64</td>
<td>_Float32x</td>
</tr>
<tr>
<td>128</td>
<td>long double</td>
<td>_Float128</td>
<td>_Float64x</td>
</tr>
</tbody>
</table>

_Float32x could have the binary128 format
• P7 – `<float.h>` macros for type characteristics
• P7 – `FLTN_IS_ARITH, DECN_IS_ARITH` macros indicate which data-interchange types are arithmetic
• P9 – usual arithmetic conversions, when formats are identical, prefer
  interchange > generic > extended
• P10 – constant suffixes
• P11 - CORRECTION: "floating or pointer" should be "arithmetic or pointer”
• P12 - non-arithmetic data-interchange types must work in conversions, classification macros, and total order functions, but not other operations
• P13 - <math.h> macro and function names for new types
• P22 – `<stdlib.h>` numeric conversion functions for new types
• P23 – `<complex.h>` functions for complex types whose corresponding real types are (binary) new types
• P25 – `tgmath` for all the new arithmetic types - same issue about equivalent formats as with usual arithmetic conversions