Title: How do you add one to something?
Author, affiliation: Aaron Ballman, Intel
Date: 2024-07-10
Proposal category: Bug fixes
Target audience: WG14 members, C implementers

Abstract: Clarifies what “appropriate type” means for the ++ and -- operators.
How do you add one to something?

Reply-to: Aaron Ballman (aaron@aaronballman.com)
Document No: N3297
Date: 2024-07-10

Summary of Changes

N3297
- Initial version

Introduction and Rationale

During discussion of WG14 N3259, which allowed ++ and -- to be used on complex types, the committee observed that “the value 1 of the appropriate type” is ambiguous. Consider an example like:

```c
unsigned _BitInt(12) bi = 0;
bi++;  
```

Is 1 of type int? _BitInt(1)? unsigned _BitInt(12)? Any of these answers is at least somewhat defensible and the standard is unclear on what we want the answer to be.

Generally, we want the type for 1 to be the same type as the type of the operand. However, special provisions should exist for:

<table>
<thead>
<tr>
<th>Type</th>
<th>Expression to yield the correct type for 1</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;sign&gt;</code> _BitInt(N)</td>
<td><code>&lt;sign&gt;</code> _BitInt(N){1}</td>
</tr>
<tr>
<td>_Complex <code>&lt;type&gt;</code></td>
<td><code>&lt;type&gt;</code>{1.0}</td>
</tr>
<tr>
<td>Pointer type</td>
<td>(int){1}</td>
</tr>
<tr>
<td>_DecimalN</td>
<td>(_DecimalN){1.DF}</td>
</tr>
</tbody>
</table>

Proposed Wording

The wording proposed is a diff from the committee draft of WG14 N3220 applied. Green text is new text, while red text is deleted text.

Add a new paragraph before the existing 6.5.3.5p2:
The adjustment value is the value used to increment or decrement the operand. If the operand has a pointer type, the adjustment value has type int and the value 1; if the operand has complex type, the adjustment value has the corresponding real type of the operand and the value 1.0; if the operand has decimal floating type, the adjustment value has the same type as the operand, 1 as the numerical value, and 0 as the quantum exponent; otherwise, the adjustment value has the same type as the operand and the value 1.

Modify the existing 6.5.3.5p2:
The result of the postfix ++ operator is the value of the operand. As a side effect, the value of the operand
object is incremented by the adjustment value (that is, the value 1 of the appropriate type is added to it).

Modify the existing 6.5.3.5p3:
The postfix -- operator is analogous to the postfix ++ operator, except that the value of the operand is decremented by the adjustment value (that is, the value 1 of the appropriate type is subtracted from it).

Modify 6.5.4.1p2:
The value of the operand of the prefix ++ operator is incremented. The result is the new value of the operand after incrementation. The expression ++E is equivalent to (E+=1), where the value 1 is the adjustment value (6.5.3.5) of the appropriate type.

Acknowledgements

I would like to recognize the following people for their help in this work: Joseph Myers