Clarify intent of P1841 numeric traits

ABSTRACT

A list of design-related questions after implementation of [P1841R2] "Wording for Individually Specializable Numeric Traits".

CONTENTS

0 Changelog 1
1 Introduction 1
2 Remaining Design Questions 1
3 Suggested Straw Polls 3
4 Straw Polls 3
A Bibliography 4
0

CHANGEOG

0.1

CHANGES FROM REVISION 0

Previous revision: P2551R0

- Removed questions that were answered in the last telecon.
- Present options for reciprocal_overflow_threshold.

1

INTRODUCTION

[P1841R2] provides wording for numeric traits. The last design paper was [P0437R1] with additions from [P1370R1]. Most of the open questions were answered in LEWG already. The question on reciprocal_overflow_threshold was deferred to let the authors of [P1370R1] and this paper determine the original intent and its consequences.

2

REMAINING DESIGN QUESTIONS

1. Decision already taken in LEWG.
2. Decision already taken in LEWG.
3. (no action requested) In the prevision discussion (and poll) of this point we noticed that the traits P2551R0 listed was incomplete/incorrect. The poll taken in LEWG therefore said: "The numeric traits that are not meaningful for numeric_limits (denorm_min, epsilon, etc) should be disabled for integral types." The resulting list then is:
   - denorm_min
   - epsilon
   - max_exponent
   - max_exponent10
   - min_exponent
   - min_exponent10
   - infinity
   - quiet_NaN
   - signaling_NaN
The (bad) list in P2551R0 was:

- denorm_min
- epsilon
- norm_min
- reciprocal_overflow_threshold
- round_error
- max_exponent
- max_exponent10
- min_exponent
- min_exponent10

4. reciprocal_overflow_threshold is currently defined as:

    template <class T> struct reciprocal_overflow_threshold<T> { see below; }

    The smallest positive value $x$ of type $T$ such that $T(1)/x$ does not overflow.

This yields a subnormal number for IEC559 types. How should this value change wrt. treat-denormals-as-zero? I.e. in a situation where the hardware treats subnormal operands as zero you get $1/0 \rightarrow \infty$, which does overflow. In which case it doesn’t match the specification anymore. This trait is specified by a behavior and as such may depend on processor state. As a compile-time constant this value must be independent from runtime behavior. But what is the correct value? See https://godbolt.org/z/eWxdnTYf8 for a demonstration of the problem.

Update after consultation with Mark and Damien (the P1370R1 authors):

- It would be possible to decouple the specification from runtime behavior by specifying behavior of constant expressions only; i.e. that $T(1)/x$ does not overflow in a constant expression.
- P1370R1 presented an algorithm to determine the value and it does not yield the "smallest positive value $x$ of type $T$ such that $T(1)/x$ does not overflow".
- The P1370R1 algorithm seems to ensure that the value is never subnormal. Thus, the specification should have been "The smallest positive normal value $x$ of type $T$ such that $T(1)/x$ does not overflow"
Since the actual reciprocal overflow threshold depends on runtime state, we’re not sure who would/should use a compile-time constant. It seems simpler and safer to remove \texttt{reciprocal\_overflow\_threshold} from P1841.

Mark wrote:

I would prefer to remove \texttt{reciprocal\_overflow\_threshold} entirely. The intent of the feature was to describe actual computer behavior at run time, so that library authors could write generic code. However, we can’t do that with traits. For example, traits can’t change value based on compiler flags. I wish I had realized that better when proposing the feature.

5. Decision already taken in LEWG.

3  Suggested Straw Polls

Poll: Remove \texttt{reciprocal\_overflow\_threshold} from P1841.

<table>
<thead>
<tr>
<th>SF</th>
<th>F</th>
<th>N</th>
<th>A</th>
<th>SA</th>
</tr>
</thead>
</table>

If the above poll doesn’t reach consensus:

Poll: Specify the behavior of \texttt{1 / reciprocal\_overflow\_threshold} only for constant expressions.

<table>
<thead>
<tr>
<th>SF</th>
<th>F</th>
<th>N</th>
<th>A</th>
<th>SA</th>
</tr>
</thead>
</table>

Poll: Require \texttt{reciprocal\_overflow\_threshold} to be a normal number.

<table>
<thead>
<tr>
<th>SF</th>
<th>F</th>
<th>N</th>
<th>A</th>
<th>SA</th>
</tr>
</thead>
</table>

4  Straw Polls

4.1  LEWG Telecon 2022-03-29

Poll: Numeric traits can deviate from \texttt{numeric\_limits}.

<table>
<thead>
<tr>
<th>SF</th>
<th>F</th>
<th>N</th>
<th>A</th>
<th>SA</th>
</tr>
</thead>
<tbody>
<tr>
<td>13</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Poll: Numeric traits should be based on representation rather than behavior (ignoring reciprocal_overflow_threshold).

<table>
<thead>
<tr>
<th>SF</th>
<th>F</th>
<th>N</th>
<th>A</th>
<th>SA</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>5</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Poll: All numeric traits for bool should be disabled.

<table>
<thead>
<tr>
<th>SF</th>
<th>F</th>
<th>N</th>
<th>A</th>
<th>SA</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>6</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Poll: The numeric traits that are not meaningful for numeric_limits (denorm_min, epsilon, etc) should be disabled for integral types.

<table>
<thead>
<tr>
<th>SF</th>
<th>F</th>
<th>N</th>
<th>A</th>
<th>SA</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Poll: max_digits10 should deviate from numeric_limits and yields digits10_v + 1.

<table>
<thead>
<tr>
<th>SF</th>
<th>F</th>
<th>N</th>
<th>A</th>
<th>SA</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>5</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

BIBLIOGRAPHY

