N2878: nsec_t && timespec::tv_nsec

timespec::tv_nsec is too wide and unsatisfiable by some architectures.
Relaxing the type by introducing nsec_t can alleviate this.
The issue

The current wording in the current draft C2X standard N2731 from 7.27.1.4 is:

```
The range and precision of times representable in clock_t and time_t are implementation-defined. The timespec structure shall contain at least the following members, in any order. The semantics of the members and their normal ranges are expressed in the comments.

time_t tv_sec; // whole seconds -- ≥ 0
long tv_nsec; // nanoseconds -- [0, 999999999]
```

However, this presents a small set of problems:

a) the minor: this is a hold-over from I16L32 architectures, and on LP64 architectures longs are way too big, and

b) the major: there are existing implementations which cannot conform to this, due to ABI requirements.

Examples

Under Linux®, on the X32 ABI, the kernel’s `struct timespec` is invariably

```
struct timespec {
    time_t tv_sec;
    long tv_nsec;
};
```

under the kernel ABI: the LP64 AMD64. This means that from the ILP32 userspace, it looks like this:

```
struct timespec {
    time_t tv_sec;
    int64_t tv_nsec;
};
```

The shortest available int64_t is long long, and the libc must expose a kernel-ABI-compatible timespec — a pickle indeed!

Proposed wording

3.1. 7.27.1.3

The types declared are size_t (described in 7.19);

```
clock_t
```

and
time_t

which are real types capable of representing times;

nsec_t

which is an implementation-defined integer type capable of representing the range \([0, 999999999]\);

struct timespec

which holds an interval specified in seconds and nanoseconds (which may represent a calendar time based on a particular epoch); and

struct tm

which holds the components of a calendar time, called the broken-down time.

3.2. 7.27.1.4

The range and precision of times representable in clock_t and time_t are implementation-defined. The timespec structure shall contain at least the following members, in any order. The semantics of the members and their normal ranges are expressed in the comments.342)

```c
    time_t tv_sec; // whole seconds -- \(\geq 0\)
    long tv_nsec; // nanoseconds -- \([0, 999999999]\)
    nsec_t tv_nsec; // nanoseconds -- \([0, 999999999]\)
```

The tm structure shall... [rest of section unchanged]

4. Rationale

Being strictly additive, this changes nothing on already-conforming implementations: nsec_t can simply continue to be long.

However, this enforces the need to cast tv_nsec to a concrete type for formatting or other processing, and allows user code to actually store it in its original form.

5. References

The current Linux ABI timespec situation: https://sourceware.org/pipermail/libc-alpha/2021-December/133702.html — this is part of a larger thread born out of an attempt to accurately describe timespec::tv_nsec as part of Linux man-pages’ system_data_types(7): https://lore.kernel.org/linux-man/ec1dce655184f6cdaaa40ff8b7970b750434e4ef.1638123425.git.nabijaczleweli@nabijaczleweli.xyz/T/

Linux® is the registered trademark of Linus Torvalds in the U.S. and other countries.
## Contents

1. The issue ........................................................................................................... 1
2. Examples ........................................................................................................... 1
3. Proposed wording .............................................................................................. 1
   3.1. 7.27.1.3 ................................................................................................. 1
   3.2. 7.27.1.4 ................................................................................................. 2
4. Rationale ........................................................................................................... 2
5. References ......................................................................................................... 2