N3066 (p1): `timespec::tv_nsec`

timespec::tv_nsec is too wide and unsatisfiable by some architectures.
Relaxing the type can alleviate this.
**N3066 (p1): timespec::tv_nsec**

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

---

1. **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.

```c
    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.

2. **Examples**

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

```c
    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:

```c
    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!

3. **Proposed wording**

3.1. **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.
```c
time_t tv_sec; // whole seconds -- ≥ 0
long tv_nsec; // nanoseconds -- [0, 999999999]
/* see below */ tv_nsec; // nanoseconds -- [0, 999999999]
```

The `tv_nsec` member shall be of an implementation-defined signed integer type capable of representing the range [0, 999999999].

The `tm` structure shall… [rest of section unchanged]

4. Rationale

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

However, this enforces the need to cast `tv_nsec` to a concrete type for formatting and other processing.

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/ec1dce6538a6f6bedae405f8b7970b750434e4ef.1638123425.git.nabijaczleweli@nabijaczleweli.xyz/T/

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