Proposal for C2Y
WG14 N3312
Title: Allowing stricter alignment for atomic types
Author, affiliation: IBM, WG14
Date: 2024-08-07
Proposal category: Technical
Reference: Base document: N3220

Atomic types may have stricter alignment requirements than their non-atomic counterparts if efficiency is needed. This can happen if the hardware supports lock-free atomic operations on objects that are aligned on larger boundaries such as 16 bytes. For example, Clang's __int128 type, atomic versions of aggregates larger than 8 bytes up to 16 bytes, as well as double _Complex types on x86_64-pc-windows and on AIX.

Requiring all objects to have the fundamental alignment as the alignment causes issues for these atomic types on those types of hardware since changing max_align_t would result in binary incompatibility with previously compiled modules. Relaxing this constraint for atomic types can allow not only efficient implementations for lock free atomics, but also allow more flexibility and efficiency for lock based atomics.

Proposed changes:

Change 6.2.8#2 from:
  – all atomic, qualified, or unqualified basic types;
  – all atomic, qualified, or unqualified enumerated types;
  – all atomic, qualified, or unqualified pointer types;

To:
  – all qualified or unqualified basic types,
  – all qualified or unqualified enumerated types,
  – all qualified or unqualified pointer types,

And append to the paragraph:
  Whether any atomic types have fundamental alignment is implementation defined.

Change footnote 49 (attached to 6.2.8#3) from:
  Every over-aligned type is, or contains, a structure or union type with a member to which an extended alignment has been applied.

To:
  Every over-aligned type is, or contains, a structure or union type with a member to which an extended alignment has been applied, or an atomic type that does not have a fundamental alignment.