Earthly Demon: Accessing a Member of an Atomic Structure or Union (Updates n3624)

Document: n3653
Author: Martin Uecker
Date: 2025-08-24

Changes since n3624: Reword to make clear only access to non-atomic structures or unions matter and remove wording about qualifiers.
Changes since n3564: Reword to not use the run-time concept of access.

Undefined Behavior: A member of an atomic struct or union is accessed.

Example:

```
    void f(void *p)
    {
        _Atomic struct foo { int x; } y;
        _Atomic int *p = &x.y;
        *p = 1;
    }
`
```

https://godbolt.org/z/qvx1485es

Analysis: This is problematic as it could lead to data races and also incorrect atomic accesses. Compilers warn about this already. It may be possible to define useful semantics in the future.

Recommendation: Make this a constraint violation.

Wording (n3550)

6.5.3.4 Structure and union members

Constraints

1 The first operand of the . operator shall have a~~n~~ **non-atomic** ~~**, qualified, or unqualified**~~ structure or union type, and the second operand shall name a member of that type.**96)**

2 The first operand of the -> operator shall have type "pointer to **non-atomic** ~~**, qualified, or unqualified**~~ structure" or "pointer to **non-atomic** ~~**, qualified, or unqualified**~~ union", and the second operand shall name a member of the type pointed to.**96)**

Semantics

~~**5 Accessing a member of an atomic structure or union object results in undefined behavior.96)**~~

96) For example, a data race would occur if access to the entire structure or union in one thread conflicts with access to a member from another thread, where at least one access is a modification. Members can be safely accessed using a non-atomic object which is assigned to or from the atomic object.

Acknowledgments: Jens Gustedt and Joseph Myers for comments.