Slay Some Earthly Demons XVII (Updates n3421)

Document: n3482 Author: Martin Uecker Date: 2025-02-15

Changes: Use the term variadic function, make the constraint stricter (as in n3363), and add alternative wording for a constraint in the library (SC22WG14.28296).

Undefined Behavior: Function takes a variable number of arguments but is not defined with ellipsis.

Example:

```
void test()
{
    extern void f(int, ...);
}

extern void f(int);
```

Godbolt Link

Analysis: The two function types would then not be compatible. This invokes a constraint violation for declarations in the same scope (6.7.1p5) or is generally UB otherwise (6.2.7p2). Thus, the specific UB is vacuous as it is either already covered by a constraint violation or a more general rule.

Recommendation: Remove.

It is further recommended to remove the implicit UB of using the `va_start` macro in a function without ellipisis.

```
#include <stdarg.h>
void test(int i)
{
    va_list ap;
    va_start(ap);
}
```

Godbolt Link

Wording (n3467)

6.9.2. Function Definitions.

Semantics

~~9 If a function that accepts a variable number of arguments is defined without a parameter type list that ends with the ellipsis notation, the behavior is undefined.~~

Additional Change Ia

6.9.2. Function Definitions.

Constraints

7 The `va_start` macro shall only be used in the compound-statement of the body of a variadic function.

Additional Change Ib (alternative: in the library section)

7.16.2.5

Synopsis

```
#include <stdarg.h>
void va_start(va_list ap, ...)
```

Constraints

2 The `va_start` macro shall only be used in the compound-statement of the body of a variadic function.

Additional Change II

6.5.3.3 Function calls

Constraints

2 **For a function that is not variadic** the number of arguments shall agree with the number of parameters. **For a variadic function there may be more arguments than parameters in the parameter type list. For each parameter the corresponding** ~~each~~ argument shall have a type such that its value may be assigned to an object with the unqualified version of the type of ~~its corresponding~~ **the** parameter.

Acknowledgements: JeanHeyd Meneide, Jens Gustedt, and Joseph Myers for clarifying comments.