JTC1/SC22/WG14 - N2907

Title: **Wording Change for Variably-Modified Types**
Author: **Martin Uecker**
Date: **2021-12-29**

This is a follow up paper to N2778 which was voted into C23. Considering the following example:

void foo(int N, int a[N]);

The parameter ‘a’ has automatic storage duration and is declared as a VLA array. The type of the parameter is then adjusted to a pointer type. For this reason, it is not a variable length array with automatic storage duration and should unconditionally be supported. But this may not be entirely clear. We suggest to add a footnote to clarify this.

Proposed Wording (relative to N2778)

6.10.8.3 Conditional feature macros
__STDC_NO_VLA__ The integer constant 1 , intended to indicate that the implementation does not support variable length arrays **with automatic storage duration.** **YYY)**

4 If the size is not present, the array type is an incomplete type. If the size is * instead of being an expression, the array type is a variable length array type of unspecified size, which can only be used in declarations or type names with function prototype scope; 146) such arrays are nonetheless complete types. If the size is an integer constant expression and the element type has a known constant size, the array type is not a variable length array type; otherwise, the array type is a variable length array type. (Variable length arrays **with automatic storage duration** are a conditional feature that implementations need not support; see 6.10.8.3.)

YYY) Parameters declared with variably length array types are adjusted and then define objects of automatic storage duration with pointer types. Thus, support support for such declarations is mandatory.