WG14 Document: N1281
Author: Konrad Schwarz
Date: 2008-03-02

C99 introduced the `_Pragma()` operator to enable creation of `# pragma` directives via macro expansion. The syntax is `_Pragma ( string-literal )`.

My enhancement request is to define `_Pragma ( string-literal-1 string-literal-2 ...)` (`_Pragma` with a white-space separated list of string literals) as a C99 `_Pragma` with an argument consisting of the concatenation of the string literals, analogouously to the concatenation of adjacent string literals performed by translation phase 6. Although this differs from the treatment of header files names by the # include directive, I offer the following reason for this request (by way of a problem that has arisen in practice):

Some pragmas require inserting arguments. E.g., the ELF object file format (and many others) allows defining one symbol as an alias of another; typical syntax is

```
# pragma alias new_name = old_name
```

This is the syntax used by at least one current compiler (from Altium/Tasking). Other compilers use slightly different syntaxes.

Dealing with an open-ended set of names, especially when encapsulating differences between compilers, requires packing up the pragma directive into a macro.

This macro might be defined as follows:

```
# define PRAGMA_ALIAS(NEW_NAME, OLD_NAME) PRAGMA_(alias
NEW_NAME = OLD_NAME)

# define PRAGMA_(ARG) _Pragma (# ARG)
```

In this case, the parameters `NEW_NAME` and `OLD_NAME` are themselves macro expanded, since they are not protected from expansion by either stringifying them with the # operator or using the ## concatenation operator on them. This may lead to unintended effects, especially if an unrelated macro definition by coincidence provides an expansion for the arguments of `NEW_NAME` or `OLD_NAME`. For macro expansions that expand to ordinary C code, the solution is to use the following sequence:

```
# define PRAGMA_ALIAS(NEW_NAME, OLD_NAME)  _Pragma ("alias
" #NEW_NAME " = " #OLD_NAME)
```

- applying the # operator prevents `NEW_NAME` and `OLD_NAME` from being further expanded. This fails in C99, since `_Pragma()` accepts only a single string.

Hence the enhancement request.