Using macros in #pragma directives

The compiler allows you to use macros in the #pragma directive.

Example:

```
#define DATA_MAC(a)  DATA_SEG a
#define CONST_MAC(b) CONST_SEG b
#define CODE_MAC(c)  CODE_SEG c

#pragma DATA_MAC(SD_PRAMA)
#pragma CONST_MAC(SK_PRAMA)
#pragma CODE_MAC(SC_PRAMA)
```

The pre-processor output (compiler switch '-Lp') has been enhanced to write out the pragma directives:

Pre-Processor output:

```
/**** FILE 'b.c' */

/*        1 */
/*        2 */
/*        3 */
/*        4 */
/*        5 */  #pragma DATA_SEG SD_PRAMA
/*        6 */  #pragma CONST_SEG SK_PRAMA
/*        7 */  #pragma CODE_SEG SC_PRAMA
```

Note

For a correct pre-processor output, it is necessary that the '#pragma' directive is not used in a macro.

Example:

```
#pragma NO_STRING_CONSTR
#define DATA_SEG_DEF(kind, short, name) #pragma kind short name
DATA_SEG_DEF(DATA_SEG, SHORT_SEG, MySeg)
```

The above example will not work correctly, because the '#pragma' is split into 2 symbols, which is not accepted by the compiler:

```
/*        1 */
/*        2 */
/*        3 */  #pragma NO_STRING_CONSTR
/*        4 */
/*        5 */  # pragma DATA_SEG SHORT_SEG MySeg
```