Macros

Las macros parecen procedimientos, pero existen sólo hasta que se ensambla el código, después del ensamblado, todas las macros se reemplazan con instrucciones reales. Si se declaró una macro y nunca se la llamó en el programa, el ensamblador simplemente la ignorará.

Una de las diferencias entre un procedimiento y una macro es que, en el programa principal, al procedimiento se lo llama con la instrucción CALL, por ejemplo, CALL SUMA. En una macro, no se usa la instrucción CALL, sino que se llama directamente a la macro por su nombre. Por ejemplo, SUMA.

Un procedimiento se encuentra en una dirección específica de la memoria y, si utiliza el mismo procedimiento 100 veces, la CPU transferirá el control a esta parte de la memoria. El control volverá al programa mediante la instrucción RET. La dirección de retorno del procedimiento se almacena en la pila. La instrucción CALL ocupa aproximadamente 3 bytes, por lo que el tamaño del archivo ejecutable de salida, al compilar el programa, crece de manera muy insignificante, sin importar cuántas veces se utilice el procedimiento.

La macro se expande directamente en el código del programa. Entonces, si usa la misma macro 100 veces, el compilador expande la macro 100 veces, haciendo que el archivo ejecutable de salida sea cada vez más grande, cada vez que se insertan todas las instrucciones de una macro.

Debe utilizar la pila o cualquier registro de propósito general para pasar parámetros al procedimiento.

Para pasar parámetros a una macro, simplemente puede escribirlos después del nombre de la macro. Por ejemplo, un procedimiento que sume dos números puede pasar lo parámetros de la siguiente manera:

Suma 2, 5

Para indicar que la macro terminó, se usa la directiva ENDM.

Por lo general, las macros se definen al principio de un programa, antes de los segmentos de datos y de código.

Al igual que los programas, las macros pueden contener datos a paritir de la directiva .data

Una macro tiene la ventaja de no tener tantos accesos a memoria como los procedimientos cuando se ejecutan las instrucciones call y ret. Un procedimiento tiene la ventaja respecto de una macro en que el tamaño del programa es menor ya que cada llamada a un macro inserta una nueva copia de las instrucciones del macro en el programa.
.

Una macro no consume tiempo de ejecución al expandirse, ya que son directivas y las directivas se aplican en tiempo de ensamblado.

Los comentarios que aparecen en las macro, aparecerán en el programa al expandirse. Para que no aparezcan se deben poner dos “;” →”;;”

Etiquetas en las macro

Las macros se expanden directamente en el código, por lo tanto, si hay etiquetas dentro de la definición de la macro, es posible que obtenga el error "Declaración duplicada" cuando la macro se usa dos veces o más. Para evitar este problema, utilice la directiva LOCAL seguida de nombres de variables, etiquetas o nombres de procedimientos. Por ejemplo:

```
crearCadena    MACRO texto
                LOCAL cadena
               .data
               cadena BYTE texto,0
ENDM
```

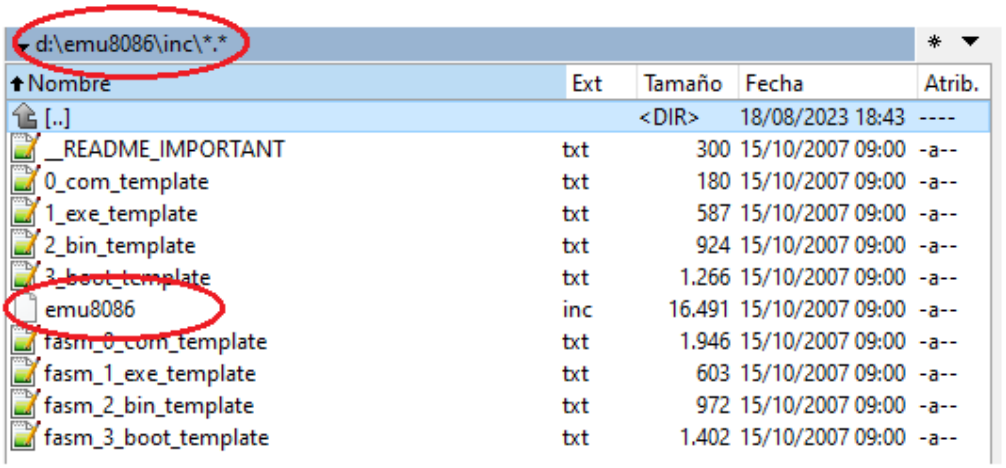
De este modo, si la macro se llama dos veces en el programa, no aparecerán dos etiquetas “cadena” iguales, lo que causaría error, ya que, ahora, la etiqueta “cadena” es local.

Include

Es conveniente poner todas las macro juntas en un archivo y, en los programas, incluir ese archivo. Por ejemplo, en la carpeta inc del emulador EMU8086, puede verse el archivo "emu8086.inc" que contiene varias macroinstrucciones.

Para invocar en un programa esas macroinstrucciones se usa, al principio del programa, la palabra reservada include:

```
include "emu8086.inc"
```



Macro recursiva

Si se requiere la expansión de una macro en forma recursiva, se hacen necesarias directivas adicionales. Cuando el preprocesador de macroinstrucciones encuentra las directivas de ensamblado condicional .if y .endif procede a abrir y cerrar una sección de ensamblado condicional en función de que se cumpla la condición.

Ejemplo de una macro

Mostrar en la pantalla la suma de 10 datos usando una macro para imprimir las unidades, las decenas y las centenas del resultado.

Archivos adjuntos:
Sin macro: ej08_sumar_10_numeros.asm
Con macro: ej08_sumar_10_numeros_conMacro.asm