Macros

Overview

A macro is similar to a procedure

A macro name represents a group of instructions

A procedure is called at execution time

control transfers to the procedure

returns after executing the procedure's statements

A macro is invoked at assembly time

Assembler copies macro's statements into the program at the point of invocation

Macro Definition and Invocation

A macro is a block of text that has been given a name

When TASM encounters the name during assembly, it inserts the block into the program

The text may consist of instructions, pseudo-ops, comments, or references to other macros

Syntax of a Macro Definition

MACRO macro_name optional-parameter-list

statements

ENDM macro_name

Example: define a macro to move a word into a word

MACRO Movw word1,word2

push [word2]

pop [word1]

ENDM Movw

Using a Macro

To use a macro in a program, we invoke it

The syntax is:

macro_name argument-list

When TASM encounters the macro name, it expands the macro

As it copies the macro statement, TASM replaces each of the formal parameters with the

corresponding actual argument

A macro must be defined before it is used

Example

Invoke the macro Movw to move B to A, where B and A are word variables

Movw A,B

To expand this macro, TASM copies the macro statements into the program at the position of the call, replacing each occurrence of word1 by A and word2 by BThe result is

push [B]

pop [A]

Caution in Use of Macros

Macros are simply text substitution -- unexpected things can happen

Example: Movw A,bx expands to

push [bx] ; register-indirect mode!!

pop [A]

Example: Movw A+2,B expands to

push [B]

pop [A+2] ; this is OK

Movw A,ax is illegal -- why?