Multimodule Programs

A large program can be divided into multiple modules (assembled units) to make it easier to manage and assemble. Each module is assembled independently, so a change to one module's source code only requires reassembling the single module. The linker combines all assembled modules (OBJ files) into a single executable file.

There are two general approaches to creating multimodule programs:

Traditional approach:

Using the EXTERN directive to declare external procedures and the PUBLIC directive to export procedures to other modules. Microsoft's advanced INVOKE and PROTO directives: simplify procedure calls and hide some low-level details.

Hiding and Exporting Procedure Names

By default, MASM makes all procedures public, permitting them to be called from any other module in the same program.

You can override this behavior using the PRIVATE qualifier or the OPTION PROC:PRIVATE directive.

PRIVATE qualifier: makes a single procedure private.

OPTION PROC:PRIVATE directive: makes all procedures in a module private by default.

You can use the PUBLIC directive to export any procedures you want.

If you use OPTION PROC:PRIVATE in your program's startup module, be sure to designate your startup procedure (usually main) as PUBLIC, or the operating system's loader will not be able to find it.

The following example shows how to create a multimodule program using the traditional approach:

```
; Module 1: mod1.asm

myProc PROC PUBLIC
; ...

myProc ENDP

; Module 2: mod2.asm

EXTERN myProc

main PROC
; ...

INVOKE myProc

main ENDP
```

To assemble the program, you would use the following commands:

This would create an executable file called myprog.exe.

The EXTERN directive tells the assembler that the myProc() procedure is defined in another module.

The PUBLIC directive tells the assembler that the myProc() procedure can be called from other modules.

Using the INVOKE and PROTO Directives

The following example shows how to create a multimodule program using Microsoft's advanced INVOKE and PROTO directives:

```
; Module 1: mod1.asm

PROTO myProc

myProc PROC PUBLIC
; ...

myProc ENDP

; Module 2: mod2.asm

INVOKE myProc

main PROC
; ...

main ENDP
```

To assemble the program, you would use the following commands:

```
ml /c /obj mod1.asm
ml /c /obj mod2.asm
link mod1.obj mod2.obj /out:myprog.exe
```

This would create an executable file called myprog.exe.

The **/c option** tells MASM to compile the source code but not link it. The **/obj option** tells MASM to generate object files. The **/out: option** tells MASM to generate an executable file with the specified name.

The PROTO directive tells the assembler about the prototype of the myProc() procedure, including its name and the number and types of its arguments.

The INVOKE directive tells the assembler to call the myProc() procedure.

The INVOKE and PROTO directives simplify procedure calls and hide some low-level details, such as the need to push and pop arguments on the stack.