

MCS—8 6

宏汇编语言参考手册

(上册)

航空部第六一八研究所情报室
航工业部

明

本手册根据美国 Intel 有限公司 1979 年最新版本《MCS-86™ MACRO Assembly Language Reference Manual No.9800640-02》译出。主要内容有：宏汇编语言概述，程序编制，定义和数据初始化、存取数据、宏代码、宏处理器语言，计算模块等八章，还有十一个附录。

本手册已根据 Intel 公司更改通知单作了增补和更改。在增补和更改部分文字的左侧用竖线作了标记。对原文中的错误也作了订正。

本手册是 Intel 公司 8086, 8088 微计算机用户的主要软件手册之一，是开发 8086 和 8088 程序的一本不可缺少的工具书。

参加本手册翻译校对的人员是：罗江鸿（译第 1～5 章）
张经伦（译第 6～8 章） 吴厚道（译附录 B, C, D） 张汝霖
（译附录 G, H, L） 赵永福（译文校对）

附录 A, E, F, J, K 按原文翻印。此外，罗江鸿、赵永福二同志承担了编辑整理工作，张经伦同志校阅过本文初稿，李自新同志对本文译稿提出了不少宝贵意见。参加本手册出版工作的其它同志也付出了辛勤劳动，在此一并表示感谢。

由于时间短促，水平有限，译文中会有不少欠妥之处，敬请读者指正。

本手册按上（1～4 章）、中（5～8 章）、下（附录）三册发行。

编 者

一九八二年十月

怎样使用本手册

本手册描述了MCS-86宏汇编语言，想给那些至少熟悉一种汇编语言，而不一定是Intel宏汇编语言的读者使用。

图0-1突出了MCS-86宏汇编语言的特点（命令，指令，操作数，标号，宏语言等）和描述这些特点的各章。详见目录和索引。

第一章图1-1表示MCS-86软件的整个开发过程，并且列出了有关手册及其序号。如果你是开发8086和（或）8088程序，你可能需要其中的大部分手册。

本手册主要是一本参考手册，这样就没有必要从头到尾阅读它。但是，为了理解这一汇编语言，你应首先熟悉下列内容：

- 语言概述（第一章）
- 分段和寻址性（第二章）
- 定义和初始化数据（第三章）
- 存取数据（第四章）

就汇编语言来说，本语言的几个特点是不寻常的：

① 它被强化分类，很像某些高级语言；数据项（标号识别代码，变量识别数据）必须按其说明来使用。第三章的表3-1提供了这些概念的摘要。

② 作为8086结构的接口，这个语言提供分段和可寻址性技巧（尤其是，SEGMENT/ENDS语句对和ASSUME命令），使你能把你自己的64K字节窗口设计到兆字节存储器中去。第二章描述分段。你必须读ASSUME命令说明和第二章的“匿名访问”说明。

③ 如果你想要定义、分配和初始化数据项，如结构、记录、数组和组合，可参考第三章。

如果你想要存取这些数据项，可参考第四章。

第五章给出了一套完整的指令记忆符。附录 J 给出了简明的摘要，包括一些例子和受影响的标志。

④ 宏处理器语言，M P L 在第七章和附录 L 中描述。不用它，你也能写程序，用它，可加速程序的开发。

最后在附录 K 中，有一个抽样程序，说明了本语言的许多独特之处。除了指令 I N 和 O U T (第五章) 外，本手册不讨论 I / O。对 I / O 布局和编程技术感兴趣的读者，可参考 MCS-86 User's Guide, No. No.9800722 和 8086 Family User's Manual, No.9800722 (它还描述了可编程双通道 8 0 8 9 及其汇编语言)。

关于 8 0 8 9 汇编语言和 ASM89 汇编程序操作的完整叙述，可参考 Intel 公司出版的 8089 Assembler User's Guide, No.9800938。

目 录

第一章 MCS-86宏汇编语言概述	1
1.1 为什么要用汇编语言写程序?	1
1.2 宏处理器是ASM86的组成部分吗?	2
1.3 汇编语言提供什么?	2
1.4 指令系统是怎样设计的?	4
1. 通用指令记忆符和宏代码	5
2. 抽样指令	6
1.5 怎样编制代码和数据?	9
1. 结 构	9
2. 数 组	10
3. 记 录	11
4. 联 接	13
5. 分段概念	13
6. 过 程	15
7. 操作数可能性	16
(1) 寄存器	17
(2) 寻址方式	18
8. 宏语言	19
第二章 编制程序	21
2.1 分段与汇编模块的关系	21
2.2 分段控制和可寻址性	22
1. SEGMENT/ENDS 命令格式	22
2. “嵌套的”或“嵌入的”段	24
3. ASSUME命令	26

~ V ~

4. 加载段寄存器	2 9
5. 段 前 缀	3 2
6. 匿名访问	3 3
7. 用匿名(分离)变量的例子	3 6
8. 串指令和存储器访问	3 6
9. 组(GROUP 命令)	3 8
10. LABEL 命令	4 0
(1) LABEL 与变量一起使用	4 1
(2) LABEL 与代码一起使用	4 2
(3) 标号可寻址性	4 2
11. 过程(PROC/ENDP 命令)	4 3
(1) 使用过程的优点	4 4
(2) 调用一个过程	4 4
(3) 递归过程, 嵌套过程和直接插入过程	4 5
(4) 从过程返回	4 5
12. 程序连接命令(NAME/END, PUBLIC 和 EXTRN)	4 7
(1) NAME 命令	4 7
(2) PUBLIC 命令	4 8
(3) EXTRN 命令	4 8
(1) EXTRN 的位置	4 9
(2) 处理外部符号的一个系统方法	5 0
(4) END 命令	5 1
13. 地址计数器(R)和 ORG 命令	5 1
第三章 定义和初始化数据	5 3
2.1 标 识 符	5 3

3.2	数据项和属性	5 3
3.3	数据定义概述	5 4
3.4	常 数	5 7
1.	允许的数值范围	5 8
2.	常数的产生	5 8
3.5	定义变量(DB, DW和 DD命令)	5 9
1.	DB, DW和DD的一般形式	6 0
2.	DB, DW和DD格式的例子	6 1
	格式①有常数表达式的初始化	6 1
	格式②定义有不肯定初始化的变量	6 3
	格式③初始化地址表达式(只对于DW和DD 和DD)	6 3
	格式④定义比2个字符长的串(只对于 DB)	6 4
	格式⑤定义和初始化一个数据列表	6 5
	格式⑥重复初始化值(DUP)	6 5
3.6	定义和初始化标号	6 6
3.7	记 录	6 7
1.	“部分”记录	7 1
2.	记录分配和初始化	7 1
3.	记录分配/初始化的例子	7 2
4.	表达式中的记录	7 3
3.8	结 构	7 4
1.	结构字段的初始(缺省)值	7 6
2.	可超控(简单)结构字段	7 6
3.	结构定义的例子	7 7

4 结构分配和初始化	7 8
第四章 存取数据(操作数和表达式)	8 1
4.1 操作数: 立即数, 寄存器, 存贮器	8 1
1. 立即操作数	8 2
2. 寄存器操作数	8 4
(1) 作为显操作数的寄存器	8 5
(2) 段寄存器	8 6
(3) 指示器和变址寄存器	8 7
(4) 通用寄存器; H 和 L 组	8 7
(5) 作为隐操作数的寄存器	8 8
(6) 标志寄存器	8 9
3. 存贮器操作数	8 9
(1) JMP 和 CALL 操作数(变量, 标号, 寄存器和地址表达式)	8 9
(2) 变量	9 3
(i) 简单变量	9 4
(ii) 变址变量	9 4
(iii) 双变址变量	9 6
(iv) 结构	9 6
4.2 属性操作符	9 8
1. 属性超控操作符	9 9
(1) PTR——指示器操作符	9 9
(2) 段超控	1 0 1
(3) SHORT 操作符	1 0 2
(4) THIS 操作符	1 0 2
(5) HIGH 和 LOW 操作符	1 0 3

2.	值回送操作符	1 0 3
	SEG	1 0 3
	OFFSET.....	1 0 3
	TYPE	1 0 5
	LENGTH	1 0 6
	SIZE	1 0 6
3.	记录特殊操作符	1 0 6
	shift-count	1 0 6
	MASK 操作符	1 0 6
	WIDTH 操作符	1 0 6
4.3	表达式	1 0 7
	操作符的等级(优先权)	1 0 7
4.4	EQU命令	1 0 9
4.5	PURGE命令	1 1 0
第五章	指令系统	1 1 3
5.1	指令和数据格式	1 1 6
5.2	指令系统百科全书	1 1 7
1.	存贮器操作数	1 1 8
2.	段超控前缀	1 1 9
3.	寄存器操作数	1 2 0
4.	立即操作数	1 2 1
5.3	指令系统的组织	1 2 4
1.	数据传送	1 2 4
(1)	通用传送	1 2 5
(2)	累加器特殊传送	1 2 5
(3)	目的地址传送	1 2 5

(4) 标志寄存器传送	1 2 6
2. 算术操作	1 2 6
(1) 标志寄存器置位	1 2 6
(2) 加 法	1 2 7
(3) 减 法	1 2 7
(4) 乘 法	1 2 8
(5) 除 法	1 2 8
3. 逻辑操作	1 2 9
(1) 单操作数操作	1 2 9
(2) 双操作数操作	1 2 9
4. 串 操 作	1 3 0
(1) 硬件操作控制	1 3 0
(2) 原语串操作	1 3 1
(3) 软件操作控制	1 3 2
5. 控制转移	1 3 2
(1) 调用, 转移和返回	1 3 2
(2) 条件转移	1 3 3
(3) 迭代控制	1 3 4
(4) 中 断	1 3 4
6. 处理器控制	1 3 5
(1) 标志操作	1 3 5
(2) 处理器停机	1 3 5
(3) 处理器等待	1 3 5
(4) 处理器脱离	1 3 6
(5) 总线封锁	1 3 6
(6) 单 步	1 3 6

第六章 宏 代 码	2 7 8
6.1 说 明 符	2 8 1
6.2 修 改 符	2 8 1
6.3 范围说明符	2 8 2
6.4 段 前 缀	2 8 3
6.5 非段前缀	2 8 4
6.6 寻址方式	2 8 5
6.7 R _{01n} 和R _{01w}	2 8 8
6.8 DB, DW和DD	2 8 9
6.9 记录初始化	2 9 0
6.10 用点操作符使参数移位	2 9 0
6.11 PROCLEN	2 9 2
6.12 指令同宏代码的匹配	2 9 3
第七章 宏处理语言 (MPL)	2 9 9
7.1 宏处理概述	2 9 9
7.2 什么是宏时间?	3 0 0
7.3 宏是什么?	3 0 0
宏扩展及付效应	3 0 0
7.4 什么是宏处理?	3 0 2
7.5 为什么使用宏?	3 0 4
7.6 参数和自变量	3 0 4
7.7 宏调用的求值	3 0 6
7.8 注解产生的宏	3 0 7
7.9 在运行时间传送字串的宏	3 0 9
7.10 调用含有实际自变量的MOVE	3 1 0
7.11 传送字节串和字串的宏	3 1 1

7.1.2	M P L 判别符	3 1 1
7.1.3	M P L 中作为串的数	3 1 2
7.1.4	表达式求值; EVAL 内部功能	3 1 2
7.1.5	算术表达式	3 1 3
7.1.6	长度功能 (L E N)	3 1 4
7.1.7	串比较器 (词法 - 关系) 功能	3 1 5
7.1.8	控制功能 (I F , R E P E A T , W H I L E)	3 1 5
1.	I F 功能	3 1 6
2.	R E P E A T 功能	3 1 8
3.	W H I L E 功能	3 1 8
7.1.9	M A T C H 功能	3 1 9
7.2.0	控制台 I / O ; 交互式宏汇编	3 2 0
7.2.1	S E T 功能	3 2 2
7.2.2	S U B S T R 功能	3 2 3
第八章	计算程式: 推荐的作法	3 2 5
8.1	建 议	3 2 5
8.2	向前访问	3 2 6
1.	变量和标号	3 2 7
2.	段	3 2 8
8.3	P L M 8 6 连接约定	3 2 8
附录 A	宏代码定义	A 1 ~ A 4 4
附录 B	存储器结构	B 1 ~ B 2
附录 C	标志操作	C 1 ~ C 5
附录 D	例 子	D 1 ~ D 4 2
附录 E	按 1 6 进制顺序编写的指令	E 1 ~ E 9
附录 F	预先定义的名字	F 1 ~ F 2

附录 G	再定位		G 1 ~ G 7
附录 H	开 题		H 1 ~ H 7
附录 J	指令系统参考数据		J 1 ~ J 1 2
附录 K	抽样程序		K 1 ~ K 5
附录 L	宏处理器语言		L 1 ~ L 1 9

第一章 MCS-86 宏汇编语言概述

本章介绍 MCS-86 宏汇编语言概况，本语言是专为编制 16 位 8086 处理器程序而设计的。图 1-1 表示 MCS-86 宏汇编程序是怎样适应于 8086 软件开发环境的。

MCS-86 宏汇编程序由下列程序生成目的模块，这些程序构成 MCS-86 软件工具系列的组成部分：

- CONV86, 它转换没有错误的 8080/8085 源文件为语法有效的 8086 源文件，并发出转换的警告和错误信息，该 8086 源文件可请求编辑。
- PLM86, 它编译用 PL/M (高级语言) 写成的程序，并生成目的模块。
- LINK86, 它把目的模块联接为加载模块。
- LOC86, 它把加载模块集中加载到绝对存储器地址。
- ICE-86, 它为 8086 提供联机仿真。

描述语言和与 MCS-86 有关的软件操作的手册，已在这本手册的序言部分，即“怎样使用本手册”中说明。

MCS-86 宏汇编语言提供了几种强有力的、使用又相当简单的编制和书写程序的手段，这个程序能在 16 位 8086 微处理器上汇编、连接、定位和执行。如果你不熟悉 8086 的设计，可阅读 8086 Family User's Manual. (No 9800722) 中有关部分。

1.1 为什么要用汇编语言写程序？

尽管高级语言 (PL/M, BASIC, FORTRAN, PASCAL) 能减少程序或系统所需要的开发时间，但它们不允许程序员接触许多芯片的特性，

比如寄存器，处理器标志和特殊的指令。况且高级语言编译和解释程序致使你的算法产生无用的代码。由于这个原因，你的汇编语言程序，虽要多花时间编码和调试，但与“等效”的高级语言程序相比，通常汇编语言程序的执行更快，且需要的存贮量更小。虽然程序开发时间本身代价很大，但是，开发时间和程序性能二者之间的取舍，应按每一种应用来具体分析。通常发现最佳的解决办法是，用高级语言写某些程序，而对时间和空间要求更严格的程序用汇编语言来写。

1.2 宏处理器是ASM86的组成部分吗？

一个ASM86调用把控制转让给MCS-86宏汇编程序包含宏处理器(作为其前端(front-end))。宏处理器扫描用宏处理器语言(MPL)写的宏定义和宏调用源文件。根据宏定义扩展宏调用，MCS-86宏汇编程序汇编产生的源汇编语言文件。

用第七章中描述的宏处理器语言(MPL)，能由汇编语言指令序列产生一个为你专用的有用的算法库。MPL和MCS-86汇编语言的结合使用，确能提高汇编语言执行时间效率和减少高级语言的开发时间。

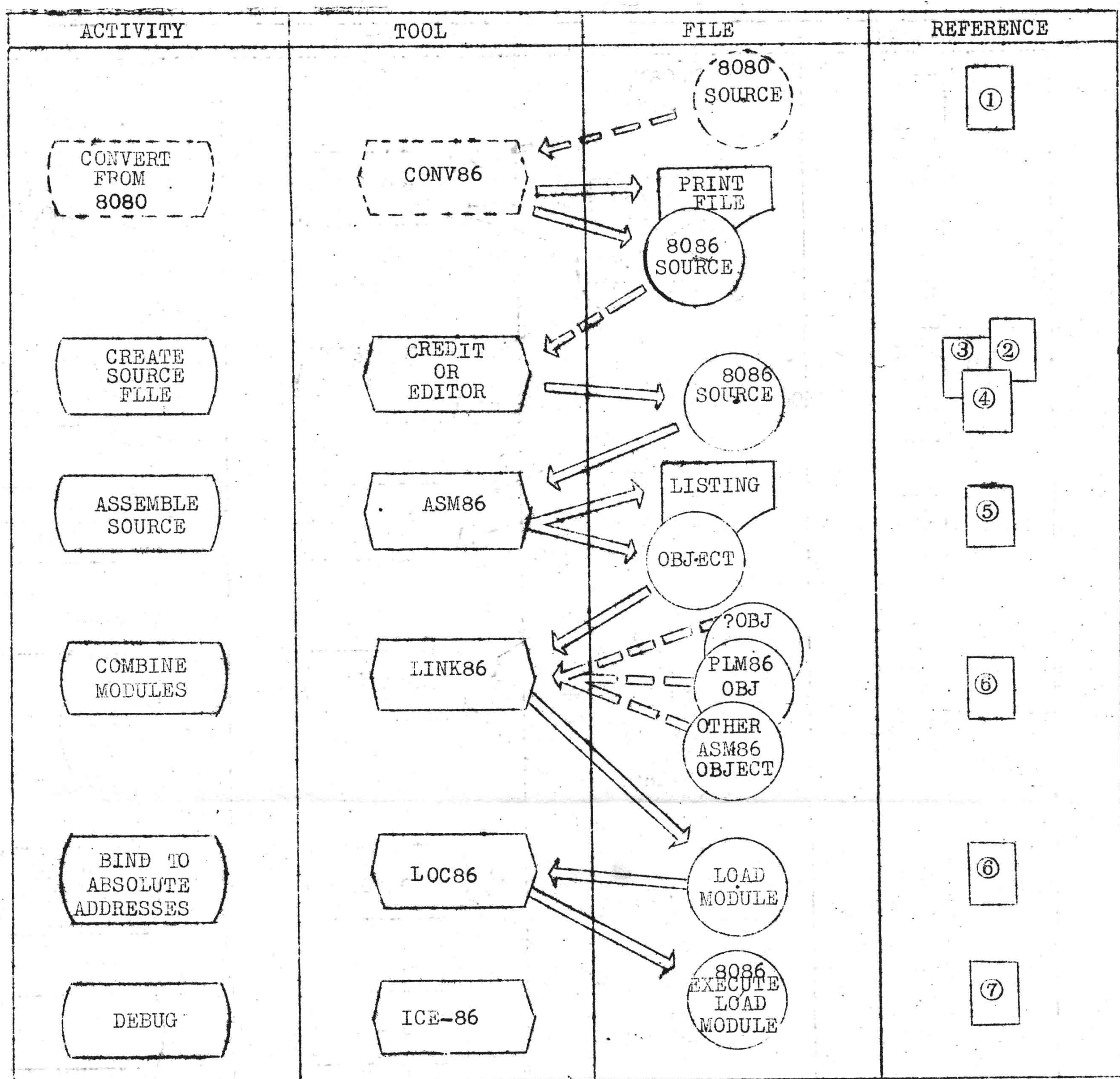
本手册中所涉及的MCS-86宏汇编语言，可参考汇编语言正文；所涉及的MCS-86宏汇编程序，可参考处理MPL和汇编语言正文的软件。

1.3 汇编语言提供什么？

汇编语言的每一个独特的优点，在这一章的后面将有进一步说明。它们包括：

- 强有力的人工设计的指令系统。
- 完善的通常只有在高级语言中才能找到的代码和数据结构技巧。
- 汇编程序检验下列内容：

① 数据使用与数据说明的一致性(这种“强化分类”(strong



- ① MCS-86 ASSEMBLY LANGUAGE CONVERTER OPERATING INSTRUCTIONS FOR ISIS-II USERS (9800642)
- ② CREDIT OPERATING INSTRUCTIONS FOR ISIS-II USERS (9800902)
- ③ ISIS-II USER'S GUIDE (9800306)
- ④ MCS-86 MACRO ASSEMBLY LANGUAGE REFERENCE MANUAL (9800640)
- ⑤ MCS-86 MACRO ASSEMBLER OPERATING INSTRUCTIONS FOR ISIS-II USERS (9800641)
- ⑥ MCS-86 SOFTWARE DEVELOPMENT UTILITIES OPERATING INSTRUCTIONS FOR ISIS-II USERS (9800639)
- ⑦ ICE-86 IN-CIRCUIT EMULATION OPERATING INSTRUCTIONS FOR ISIS-II USERS (9800741)

图 1-1. 软件开发与 MOS-86 宏汇编语言

typing)用在大多数高级语言中)。

② 保证产生最短的有效形式的指令形式。

- 宏语言可与任何串操作语言相媲美。

1.4 指令系统是怎样设计的？

大多数汇编语言在指令记忆符(如: MOV, ADD)和操作码(一般按二进制给出)之间定义了一个“一一对应”的关系。MCS-86宏汇编语言建立了一对多的对应关系;一条指令记忆符可汇编为几种操作码中的一种,取决于所提供的操作数(数据项)的类型。然后,汇编程序根据操作数类型,选择适当的操作码。

因此,MCS-86宏汇编语言是一种“强化分类”的语言:

- 分类的原因是数据可以说明为具有不同的形式(如:字节,字,双字)。
- 强化分类的原因是在同一操作中不允许有混和的操作数类型(例如,传送一个已说明的字节到一个字寄存器,或者反之)。

强化分类防止你在下列问题上疏忽大意:

- 传送一个字到字节目标操作数,因而修改了一个相邻的字节。
- 传送一个字节到字目标操作数,因而把一个无意义的数据留在相邻的字节里。

但强化分类并不妨碍你有意地执行这些类似的操作。例如,如果说明 DATA8 为字节类型, DATA16为字类型,且当:

- 你想要把 16 位寄存器 AX 的内容传送到 DATA8,因而修改了 DATA8+1 的内容,则你编码:

```
MOV WORD PTR DATA8,AX;Move AX to word at DATA8
```