MCS-51编辑、扩展汇编使用手册

复旦大学计算机系微机开发应用研究室

MCS—51 编辑扩展汇编使用手册

一、简　介

MCS—51 是一种高性能 8 位单片机，适于用作工业控制及仪器仪表的控制器。由于这些应用场合中对速度要求很高，所以一般需用汇编语言来编制程序，同时这些应用场合中常需要采用浮点计算。为了满足这些方面的要求，我们编制了FD —ASM51 扩展汇编程序，并把它配在FD — SRC — 51 单板机和SICE 仿真器里。

FD — ASM51 是MCS — 51 自汇编程序，它能把MCS—51 的汇编指令翻成机器码；它允许使用MCS—51 的特殊功能寄存器名和特殊殊功能位名；可使用十进制或 16 进制数；允许使用地址标号及符号数据；具有ORG、DB、DW、EQU、EQW、BIT 等各种伪指令。另外，FD—ASM51 具有宏子程序调用指令，允许用使用 CALL 扩展指令来调用并自动生成各种浮点运算子程序，包括加、减、乘、除、正弦、开方、多项式运算、浮点二翻十等。这使它成为独一无二的具有浮点运算功能的扩展汇编系统。

为了产生能够输入、修改、存贮用户的汇编程序我们是编制了与FD—ASM51 配套的编辑程序—— FD—EDIT。

与FD—SBC—51 单板机相配的FD — ASM51 与FD—EDIT 程序装在一块编辑汇编扩展板上，它必须与Monitor51 V2.2 版本的监控连用。板上面有 2 个EPROM 插座，一个为FD—ASM51 与FD— EDIT 程序另一个为备用插座供今后扩展用。另外，板上还装有32K RAM，用作源程序存贮器及汇编工作区，它的地址为8000H — FFFFH。使用编辑汇编扩展板时，用户不能使用8000H — FFFFH 的程序空间及数据空间。但在完成编辑和汇编功能后，用户可把得到的目标程序装入 2000H — 3FFFH 或 4030H — 5FFFH空间的RAM 或EPROM 中，这时可拔掉编辑汇编扩展板，接上用户自己的扩展板，这样用户即可使用 8000H—FFFFH 的空间。

SICE是单片微机在线仿真器的英文名称Single Chip Microcomputer In Circuit Emnlator的缩写。它是我们最新研制的一种通用单片微机在线仿真器，把它与终端或任何带有RS－232串行通讯口的计算机相连，即可构成一台单片微机开发系统。在SICE中，我们配有FD—EDIT 编端程序和FD—ASM51、FD—ASM48汇编程序。SICE内共有48K RAM，可用作源程序存贮器、汇编及系统工作区（约2K）及目标程序存贮器。

二、FD－EDIT 编辑程序

FD—FDIT 是一种具有行编辑功能和源程序存贮功能的编辑程序，它的编辑功能和编辑命令格式与IBM—PC的EDLIN行编辑程序基本相同，而源程序存贮与其他功能则是它所特有的。

1．编辑状态进入方法：

在监控状态下（提示符为“ ），输入ED↵即转到FD－EDIT 编辑状态，出现：

```
FD－EDIT V1.0
Copyright 1986 Microcomputer Lab.Fudan University
>
```

这里的“＞”为编辑状态的提示符。

如果已经输入过源程序，以后又通过EXIT命令退出编辑回到监控状态，并且没有破坏过RAM中的源程序。这时，如果想再次进入编辑状态，并且不想破坏已经输入的程序，可使用再次进入编辑状态的命令。

```
ED␣A↵
```

这时，出现提示符“＞”，表示已进入编辑状态，可对上次已输入在RAM（源程序存贮器）中的源程序进行修改、列表、汇编等操

作。

如果在再次进入时，不小心错按了ED /，则编辑的指针全部置为初态值（空），这时，如想恢复原来的源程序，必须按下FD—SBC—51单板机或SICE仿真器上的复位按钮，以回到监控状态，然后再用ED␣A /命令进入编辑状态，这样就能恢复已输入的源程序，但是如果在错按ED后用EXIT命令退出编辑回到监控，则前次输入的源程序再也无法恢复了！

在刚加电后，第一次进入编辑状态时，必须使用ED /命令，否则将产生无法预料的后果。

2．源程序存放格式：

FD—EDIT能接收大写或小写字母，对于小写字母则把它转换成大写字母，再存放到存贮器中。

FD—EDIT采用8031的外接数据存贮器作源程序存贮器。该存贮器区域一般只有32K，这与IBM—PC等个人计算机相比是相当小的，故必须尽量减小源程序的存贮空间。为了达到这个目的，在存放用户输入的源程序时，FD—EDIT把它们先进行适当的压缩，然后再存存入源程序存贮器中。

这里采用了二种压缩技术：

① 去掉不必要的空格；对于在用户输入的汇编指令中的空格符，又要不是用作分隔符的，就自动予以删除。例如对于指令

ABC:␣␣␣␣MOV␣␣␣A，#10 /

ABC为标号，冒号后的四个空格为不必要的，MOV为指令码，它后面至少应有一个空格，但也只要有一个空格就足够了，故后二个空格也为多余的。这样实际存入存贮器的指令为

ABC: MOV␣A，#10 /

共省了6个字节。

在列表或打印输出时，FD—EDIT能自动插入空格，以使显示的汇编源程序清晰易读。

② 去掉注解：由于存贮空间的限制，用户输入的注解（字符';'后至回车前的一切字符），都不存入存贮器。

采用以上二个方法，使得源程序容量与目标程序容量之比可达6：1左右，即30K源程序，可生成5K左右目标程序。再加上FD—ASM51可自动生成浮点运算子程序，这样实际上的目标程序为6K左右。对于大于6K的目标程序，可采用分段输入，分段汇编的方法，使用ORG来决定每段的目标程序开始地址，用EQW来定义后段程序用到的与前段程序有关的标号值，这样可完成较大系统的汇编程序的输入及汇编。

在存放源程序时，不存放行号。行编辑程序编辑时，能自动搜索、生成源程序的行号。

3。行编辑命令

以下命令格式中大写字母为命令字符，小写字母n及n_1为行号，它为1—9999之间的数字。

① 插入命令I：它用于在指定行号处插入一段源程序。

I ／在已输入源程序末插入。如没有输入过源程序，则从第一行　　行开始输入源程序。

nI ／在第n行前插入。

打入I命令后，进入插入状态。先显示插入行的行号。这时可输入一行源程序。输入时可用＜DEL＞（删除键）或＜BS＞（©H）键删除已输入的一个字符，也可用©X删除已输入的一行程序（在打入回车键前）。一行输入完毕，打入回车键，这时该行程序就写入到源程序存贮器中。然后行号加1，显示新的行号，可再次插入一行。插入完毕，打入＜ESC＞键，可退出插入方式，回到行编辑状态。

例1：输入一段只有二条指令的源程序，先打入I ／

显示　1

这时输入　ABC:MOV　A，#10 ／

显示　2

这时再输入　MOVX @DPTR，A ／

显示　3

打入<ESC>，退出行编辑，显示

>

② 列表命令L：它用于在显示器上列表显示指定的源程序。

L ↵　从第一行开始，至最后一行，显示全部的源程序，在显示时可用Ⓒ S暂停显示行的转动，以后打入任何字符可重新开始显示后面的程序。用Ⓒ C可停止命令的执行，回到行编辑状态。

nL ↵ 显示第n行源程序。

n_1, n_2 L ↵显示从第n_1行到第n_2行之间的源程序。

每行的显示格式如下：

行号（1－5个数字）␣标号（1－7个字符）：指令码（2－5个字符）␣操作码，凡字符数小于最大值时，均填入空格。

例2：在执行完例1输入程序后打入L ↵ 屏幕上显示：

```
>L
1        ABC:  MOV   A, #10
2              MOVX  @DPTR, A
>
```

③ 打印命令P：它用于在外接的打印机上打印指定的源程序。

P ↵　打印全部源程序。

nP ↵　打印第n行。

n_1, n_2 P ↵ 打印从第n_1行到第n_2行之间的源程序。

在执行P命令前，必须在指定的打印口上外接打印机，并接通电源，否则执行P命令将引起死循环，这时只有用复位才能使FD—SBC—51或SICE重新运行。

④ 行修改命令　n ↵：它用于对已输入的一行源程序进行修改。在打入行号n及回车后，显示该行原来程序的内容（格式同nL命令），然后在该行下面显示行号n，以后可对该行进行修改。

这时可用Ⓒ A抄录原来的字符，需修改处可打入新字符。修改完

毕，打入回车，回到行编辑状态。

在显示行号后，如不需要修改，可直接打入回车。

⑤ 删除命令D：它用于删除已输入的指定源程序。

D ↵ 删除现行行。

nD↵ 删除第n行。

n_1，n_2D 删除从第n_1行第n_2行之间的全部源程序。

⑥ 字符串搜索命令S：在指定行中搜索指定字符串。

Sl ↵ 在令P源程序中搜索字符串l。

nSl↵ 在第n行中搜索字符串l。

n_1，n_2Sl 在第n_1行到第n_2行之间搜索字符串l。

每次找到字符串l时，显示该字符串所在行的内容，然后讯问

Continue（Y／N）?

需继续搜索时，可打入Y，不需要搜索时，打入N或其他字符。

在找不到该字符串l时，显示Not Found。

例3：设已输入四条指令。

```
1    ABC:   MOV    A,#10
2           MOVX   @DPTR,A
3           INC    DPTR
4           DJNZ   R0,ABC
```

现输入命令　　SABC ↵

即在这段源程序中搜索字符串ABC。

执行该命令后，显示

```
1    ABC:   MOV    A,#10
```

Contfnue（Y／N）?

如要再向下搜索，打入Y，这时显示

```
4           DJNZ   R0,ABC
```

Continue（Y／N）?

如果打入Y，则显示

Not Found，回到编辑状态。

⑦ 字符串替换命令R：替换指定行内的一个字符串。

Rl_1／l_2　把全部源程序中的字符串l_1换成字符串l_2。命令中中间的'／'为二个字符串中的分隔符。

nRl_1／l_2　把第n行中的字符串l_1换成字符串l_2。

n_1，n_2 Pl_1／l_2　把第n_1行到第n_2行之间的字符串l_1换成字符串l_2。

例4：对于例3中的四行源程序，现在想把所有标号ABC换成LOOP，可打入

RABC／LOOP ↵

这样就完成了把所有的标号ABC变成LOOP。

注：在S，R命令中，打符串l中不允许有字符'／'。

注：在执行以上7种编辑命令时，如找不到命令指定的行号时，显示 Not Fouud。

4．编辑控制命令：

在把FD—SBC—51或SICE与终端相连时，可以用录音机作外存设备。可使用CSV命令来把内存中的汇编源程序写入磁带中。也可使用CLD或CLA命令来读入源程序。

如果把FD－SBC—51或SICE与带有磁盘的个人计算机如IBM—PC，APPLE—II等相连，同时配上这些个人计算机用的通讯控制程序，则可以使用它们的磁盘作为FD—EDIT的源程序外存设备。这时可使用MSV、MLD、MLA等命令来读写源程序。

① 源程序写带命令：CSV ↵

用它可把内存中的全部汇编源程序以16进制数形式转存到外接的盒式录音机中。转存时，先输出源程序存贮区的首地址和末地址，然后输出源程序，最后为一个字节的检验和。同时在终端的屏幕上显示写入磁带上的数据的16进制代码。磁带机操作方法同FD—SBC－51或SICE的程序写带方法。

② 源程序读带命令，CLD ↵

执行时，先清除源程序存贮区，然后把磁带上的源程序读入到存贮区中。在读入时，在终端的屏幕上显示读入的数据的16进制代码。

（格式同写带命令）。

CLA /

执行此命令时，不清除源程序区，而是把磁带上的源程序放于源程序存贮区的空闲单元，这样可把磁带上的源程序接在原来的程序的后面。

③ 源程序记盘命令：MSV©V

此命令不是以回车结束的，而是以©V结束的。在连接IBM—PC并使用TERMSBC通讯软件时，打入该命令后，主机讯问：

New Source File Name:

要求输入该源程序的文件名（以回车结束）。文件名的格式必须符合主机（IBM－PC）操作系统的要求。

打入文件名后，开始向主机传送源程序，一面传送，一面在显示器上显示传送内容。传送完毕，主机把全部接收到的源程序记入盘中。以后可象其他文件一样，用DOS或IBM－PC的编辑程序对它进行修改或打印等操作，还可用IBM—PC上的MCS—51交叉汇编对它进行汇编。

注：用CLD命令读入源程序，再用MSV命令存入主机，可实现把磁带上的汇编源程序转存到磁盘上，反之亦然。

④ 源程序读盘命令：MLD©W

此命令不是以回车结束的，而是以©W结束的。在连接IBM—PC并使用TERMSBC通讯软件时，打入该命令后，主机讯问。

Output Source File Name:

要求输入将传输的源程序的文件名（以回车结束）。该文件可以是以前用MSV命令记盘的，也可以是用IBM—PC的编辑程序所生成的汇编源程序。

打入文件名后，先清除源程序存贮区，然后主机把磁盘上的源程序传输到FD—SBC—51或SICE的源程序存贮区中，同时在屏幕上显示传输内容。

MLA ©W

此命令操作方式与MLD基本相同．只是它不清源源程序存贮区，而是把磁盘上的源程序传输到原来的源程序后面。

⑤ 源程序存贮区大人检测命令： CK↵

显示源程序存贮区的下一个空闲单元地址。

注：

FD－SBC—51中，源程序存贮区的首地址为8600H，末地址为0FFFFH。

SICE中，源程序存贮区的首地址为0730H，末地址取决于所需的用户目标程序存贮区（出借RAM）的首地址，一般为7FFFH，最大为0BFFFH。

⑥ 源程序存贮区清除命令：CLR↵

它通过清0源程序行数指针和源程序存贮区末地址指针，达到清除源程序存贮区的目的。

⑦ 退回监控状态命令：EXIT↵

执行该命令，保存现在的源程序指针，然后从编辑状态退回到监控状态，这时提示符变为'*'。以后可打入监控命令。

注：通过按FD—SBC—51或SICE的复位键，也可直接退回到监控状态（需再打入一个回车键），这时不保存现在的源程序指针，存放的为上一次执行EXIT时保存的源程序指针。

注2：在使用FD—SBC—51时，如果用EXIT退回到监控状态后，再按复位键。这时，在运行用户程序时，可使用8000H以上的程序存贮器及数据存贮器空间，也不会破坏RAM中的源程序。（在用户不使用8031的P3.5脚时）。

⑧ 汇编命令：

1）ASM↵

打入该命令后，对源程序存贮区中的汇编程序进行汇编。汇编后的目标程序存放于由ORG指令所指出的RAM单元中。

执行时，先显示：

FD ASM—51 V1.0

Copyright 1986 Microcompater Lab Fudan

University

Pass 1

开始对源程序进行第一遍扫描。在这遍扫描中，生成用户符号表，并对程序进行语法检查。

第一遍扫描中，如有错误则显示出错信息，扫描完，显示出错数目，然后返回编辑状态。这时可对源程序进行修改。

如没有错误，则讯问是否需要列表显示汇编结果：

Display List ? (Y／N)

始回答 ' Y '，则再讯问是否需要打印汇编结果：

Print List? (Y／N)

如需打印，可打入 ' Y '（这时应接通打印机）；不需打印，可打入 ' N '，或直接打入回车。

接着开始第二遍扫描。如回答需要列表显示汇编结果的，则一面汇编，一面在显示器上逐行显示汇编的结果（可用Ⓒ D来把屏幕上的信息记盘）。如不要求显示，则只进行第二遍扫描。在第二遍扫描中，逐行生成机器码，放入由ORG命令，所指示的RAM单元中（没有ORG命令时为从0000开始）。

在第二遍扫描完成后，如有错误，则显示出错内容，然后回到编辑状态。如没有错误，则显示用户符号表（这时可用Ⓒ D命令来执行显示记盘操作（接主机时），或用Ⓒ P命令来打印用户符号表（外接打印机时））。然后显示RAM中目标程序的下一个空闲单元地址，再返回编辑状态。在使用扩展宏子程序调用指令CALL时，由于在用户的目标程序后自动插入了所调用的各种浮点运算子程序，故这时显示的RAM的下一个空闲单元地址将远大于用户自己程序的末地址。

2) ASM n /

这里n是十进制或16进制数（以数字开头，以H结尾），它是目标程序的存放地址。

这个汇编命令与ASM /基本相同，又是后者目标程序放在由ORG伪指令指定的单元中，而前者目标程序存放在由n指定的单元中，其

他操作完全相同。

例5：对下面的一段源程序。

```
        ORG  0000H
        AJMP START
        ORG  0003H
        AJMP INT0
        ORG  0040H
START:  MOV  A, #5
        SETB EA
        SETB EX0
HERE:   SJMP HERE
INT0:   MOV  P1, A
        RETI
```

如打算在FD—SBC—51上汇编这段程序，并写入EPROM中，由于地址0000—1FFFH为监控程序区，没有RAM，故无法把目标程序生成在由ORG指令的区域。这时，如在地址2000H处插有RAM，可用

```
ASM  2000H ↵
```

命令对它进行汇编，汇编后生成的机器码放在从2000H开始的单元中。即AJMP START的目标指令在2000H、2001H中，AJMP INT0的目标指令在2003H、2004H中，START后的目标指令在2040H开始的单元中……。2002H及2005H—203FH空，不改变它们原来的值。

对于FD—SBC—51，由于0000—1FFFH为监控区，4000H—402FH为工作区，8000H—0FFFFH为源程序存放区，如用ASM 汇编时，ORG指出的地址空间不应与它们重合。如用ASM n 汇编时，则ORG的地址可任意，但n开始的存放目标程序的空间不能与上述地址空间重合。

对于SICE，由于它具有二种汇编：FD—ASM51及FD—ASM48，所以在汇编命令中必须指出是对MCS—51还是MCS—48的源程序进行汇编，故汇编命令分别改为：

ASM51 /及 ASM51 n /

其他操作与FD—SBC—51相同。

另外，SICE中，用户程序应放在从8000H开始的区域中，故用ASM51 /汇编时，ORG的地址必须大于等于8000H；用ASM51 n /汇编时，ORG地址可任意，但n应大于等于8000H。

注意：在使用ASM n 或ASM51 n /时，如果n不等于源程序第一条ORG指令的地址，则一般情况所生成的目标程序不能直接进行运行调试，只能用于写EPROM或暂存用。如要调试，则应把所生成的目标程序搬至由ORG所指定的单元中。

FD—ASM51是一个高速的汇编系统，它比IBM—PC上的MCS—51交叉汇编速度要快几十倍，而且使用方便（只需打入ASM）。在使用时。一般情况第一次对某源程序进行汇编时，应用不列表显示方式，在没有错误后，再用列表显示方式来进行汇编，这样可加快程序编辑、排错速度。

三、FD—ASM51汇编程序

FD—ASM51是一种具有浮点运算功能的MCS—51扩展汇编系统。下面分别介绍它的数据：指令格式。符号和伪命令等使用方法。

1．数据格式：允许使用十进制数或16进制数，后者必须是以数字开头，以H结尾，否则会出错。

2．特殊功能寄存器名：允许使用特殊功能寄存器名或直接地址来在指令中表示特殊功能寄存器，其对照表如下：

特殊功能寄存器名	16进制地址	十进制地址	特殊功能寄存器名	16进制地址	十进制地址
ACC	0E0H	224	RC2H (RCAP2H)	0CBH	203
B	0F0H	240	RC2L (RCAP2L)	0CAH	202
DPH	83H	131	SCON	98H	152
DPL	82H	130	SP	81H	129
IE	0A8H	168	T2CON	0C8H	200
IP	0B8H	184	TCON	88H	136
P0	80H	128	TH0	8CH	140
P1	90H	144	TH1	8DH	141
P2	0A0H	160	TH2	0CDH	205
P3	0B0H	176	TL0	8AH	138
PCON	87H	135	TL1	8BH	139
SSW	0D0H	208	TL2	0CCH	204
SSBUF	99H	153	TMOD	89H	137

3. 位表示法：

对于RAM中的位可寻址位必须用直接地址表示（可用十进制或16进制数），对特殊功能寄存器中的位可寻址位可用三种形式表示：①位名 ②特殊功能寄存器名。位地址 ③直接位地址，见下表。

位名	特殊功能寄存器位	位地址（16进制）	位名	特殊功能寄存器位	位地址（16进制）
CY	PSW.7	0D7H	PT2	IP.5	0BDH
AC	PSW.6	0D6H	PS	IP.4	0BCH
F0	PSW.5	0D5H	PT1	IP.3	0BBH
RS1	PSW.4	0D4H	PX1	IP.2	0BAH
RS0	PSW.3	0D3H	PT0	IP.1	0B9H
OV	PSW.2	0D2H	PX0	IP.0	0B8H
P	PSW.0	0D0H	SM0	SCON.7	9FH
TF1	TCON.7	8FH	SM1	SCON.6	9EH
TR1	TCON.6	8EH	SM2	SCON.5	9DH
TF0	TCON.5	8DH	REN	SCON.4	9CH
TR0	TCON.4	8CH	TB8	SCON.3	9BH
IF1	TCON.3	8BH	RB8	SCON.2	9AH
IT1	TCON.2	8AH	TI	SCON.1	99H
IE0	TCON.1	89H	RI	SCON.0	98H
IT0	TCON.0	88H	TF2	T2CON.7	0CFH
EA	IE.7	0AFH	EXF2	T2CON.6	0CEH
ET2	IE.5	0ADH	RCLK	T2CON.5	0CDH
ES	IE.4	0ACH	TCLK	T2CON.4	0CCH
ET1	IE.3	0ABH	EXN2	T2CON.3	0CBH
EX1	IE.2	0AAH	TR2	T2CON.2	0CAH
ET0	IE.1	0A9H	CT2	T2CON.1	0C9H
EX0	IE.0	0A8H	CP2	T2CON.0	0C8H
RD	P3.7	0B7H		P1.7	97H
WR	P3.6	0B6H		P1.6	96H
T1	P3.5	0B5H		P1.5	95H
T0	P3.4	0B4H		P1.4	94H
INT1	P3.3	0B3H		P1.3	93H
INT0	P3.2	0B2H		P1.2	92H

位名 位名	特殊功能 寄存器位	位地址 （16进制）	位名	特殊功能 寄存器位	位地址 （16进制）
TXD	P3.1	0B1H	T2EX	P1.1	91H
RXD	P3.0	0B0H	T2	P1.0	90H
	P0.7	87H		P2.7	0A7H
	P0.6	86H		P2.6	0A6H
	P0.5	85H		P2.5	0A5H
	P0.4	84H		P2.4	0A4H
	P0.3	83H		P2.3	0A3H
	P0.2	82H		P2.2	0A2H
	P0.1	81H		P2.1	0A1H
	P0.0	80H		P2.0	0A0H

4. 符号：

用户定义的符号必须是以字母@、A、……Z开头，后接数字或字母的字符串。符号中不允许使用非数字或非字母的字符。FD-ASM51能认识的字符串长度最长为4个字符，最多允许255个用户符号（包括浮点运算子程序名），超过时要出错。

用户符号可用作程序标号、数据代号等。

在SICE中，保留汇编时所生成的用户符号表。在调试程序时，可按源程序中的标号进行启动、设断点等操作，与目标地址无关。

例如对于例5的源程序，我们打算从START处开始启动程序，运行到遇到产生外部中断0，转到中断处理程序入口处停止执行，可打入如下运行控制命令，GO START，INT0

即可实现上述要求。如果以后在START前或程序任意地方又插入了许多指令，该控制命令仍然有效。这种功能叫做符号化调试功能，有了它，将实现在汇编源程序的基础上的调试而不是基于16进制目标地址调试，从而大大方便调试，加快调试进度。

5．伪指令：

ED-ASM51允许使用ORG、EQU、EQW、BIT、DB、DW等6种伪指令，前4种在汇编时不生成目标指令。

① 定位伪指令：

格式 ORG m ↵

m可为十进制或16进制数。m指出在该伪指令后的指令的汇编地址。对于用ASM↵或ASM51↵进行的汇编，m也为以后生成的目标指令的存放开始地址。对于用ASM n↵或ASM51 n↵进行的汇编，如ORG m↵为源程序的第一条指令，则m为汇编地址，m为存放地址。如ORG m↵不是源程序的第一条指令，而第一条指令的汇编地址为x（由第一条ORG X↵伪指令决定，如第一条不是ORG伪指令，则X＝0），则后继指令的目标指令的存贮地址＝n+m−x。

② 定义字节伪指令：

格式 DB x_1，x_2 …，x_n ↵

X_i为单字节数据，它为十进制或16进制数，也可为由EQU定义的符号，这时，每一个x_i对应于一个字节。

X_i也可为由两个单引号'所括起来的一个字符串（不允许含有单引号，无小写字母），这时该X_i定义的字节长度等于字符串长度，每个字节对应于一个字符的ASCII码。

③ 定义字伪指令：

格式 DW y_1，y_2，……，y_n ↵

y_i为双字节数据，它可为十进制或16进制数，也可为程序标号，或由EQW定义的符号，每一个y_i对应于二个字节。

④ 数据字节赋值伪指令：

格式 X EQU n ↵

x为用户定义的符号，n为十进制或16进制数，也可为特殊功能寄存器名字。它把n的值（单字节）赋给x。

x可用于指令的单字节操作数，包括立即数据data和直接地址direct。