㈡ SASM 汇编器

1.SASM 汇编器命令选择

-e×××	设置最大的错误输出个数（缺省为 15 个）
-E×××	错误信息输出到文件名为×××的文件
-m	设置列表文件中的宏扩展行（缺省值为 64 行）
-s	在目标文件中输出 SLO 格式的调试信息
-v	显示版本信息

2.输出文件　.hex　可执行程序代码，二进制文件
　　.SIT,DBG　适用于调试环境文件
　　.LST　源文件和程序代码文件
　　.MAP　段和全局符号表信息

3.源文件编写

一个完整的指令语句包括 4 个部分：

（标号）　指令　操作数　（注释）

标号区分大小写，指令、伪指令、条件转移代码、工作寄存器不区分大小写。标号可以用字母、数字或下划线，但起始位置必须为字母或下划线。一个程序行最长为 255 个字符，超过的字符将被忽略；标号最长为 31 个字符，标号在顶行写时，可不要冒号；不顶行时，要冒号。伪指令开始可以用‘ . ’；注释行用‘ ; ’与指令分开。

4.SASM 伪指令

ORG/ORIGIN　定义程序起始地址

EQU/EQUAL/EQUATE　定义符号值

```
HOURS:    .EQUATE   24
 MINS:    .EQUAL    60
NUMBER:   .EQU      65536*2/128
 EQU1:    .EQU      EQU2      ;错误
 EQU2:    .EQU      0
 EQU3:    .EQU      EQU2      ;正确
```

SEC/SECTION　定义数据段或程序段,PROGRAM 定义程序体，DATA 定义数据段

格式：　SECTION　OPTION
　　SEC

例：

```
　　ORG     100H
　　ADD     R0,#0
PROGRAM: .SECTION          ；定义程序段名
   DATA: .SEC              ；同上
　　ADD     R0,#0
　　.DATA                  ；定义数据段
　　DW      1234H
　　.PROGRAM               ；定义程序段
　　ADD     R0,#0
```

GLOBAL/PUBLIC　全局标号定义

EXTERN/EXTERNAL　外部符号定义

DEFVAR　　定义并初始化标号；如果没有初始化用 DEFVAR 定义的变量，将会被初始化为 0。用 DEFVAR 定义的变量值可以用 SETVAR 再次设定变量的值；而用 EQU 定义的变量则不可以，其值是恒定的。

格式：　.DEFVAR　[=NUM,] |　;
　　.SETVAR　用于给.DEFVAR 定义的变量重新赋值

例：
```
.DEFVAR   =10H,A,B,C,D   ;定义 A，B，C，D 并初始化其值为 10H
 LD       R0,#A          ;R0=10H
 A        .SETVAR  20H
 LD       R0,#A          ;R0=20H
```

例：
```
YEAR      .EQU     1998
          .DEFVAR  =YEAR,DATE
DATE      .SETVAR  DATE-1     ;DATE=1997
DATE      .SETVAR  DATE+2     ;DATE=1999
```

VECTOR　　定义中断向量地址

格式：
```
VECTOR    VECTOR_ADDR,SERVICE_ADDR    ;(SAM8)
VENT      EMB,ERB,SERVICE             ;(SAM4)
```

.BYTE/DB　　定义 8 位数据

```
        |   .DB     n,……
        |   .BYTE   n,…….
```

例：
```
DATA1   .DB     1,2,3
DATA2   .DB     11,12,13
```

在 SAM8 中
```
        LDC     R0,DATA1
        LDC     R1,DATA2+1
```

WORD/DW　　定义 16 位数据

```
    |   .DW     n,……
    |   .WORD   n,……
```

例：
```
DATA1     .DW     1,2,3
DATA2     .DW     1234H
```

DL/LONG　　定义 32 位数据

```
    |   .DL     n,……..
    |   .LONG   n,……..
```

例：
```
DATA3     .DL     1,2,3
DATA4     .LONG   9ABCH
```

.ASCII/ASCIZ　　定义字符串。ASCII 不在最后产生空格，ASCIZ 在最后产生空格

例：
```
TITLE     .ASCII    “SAM8 test program” ;不产生空格
TITLE     .ASCIZ    “I am”, ”aby”       ;在每句后面产生空格
```

.FLOAT　　把值转化为 32 位数据

.DOUBLE　　把值转化为 63 位数据

.BLKB/BLOCK　　保留几个字节的存贮空间。如果指定有值，把相应的值填入此空间；如果没有指定的值，则把 0 填入此空间。

格式：|　　.BLKB　　n[,value]

|　　.BLOCK　n[,value]

```
.BLKB     10          ;10 字节的 0
.BLOCK    4,33H       ;4 字节的 33H
```

BLKW　　保留几个字的存贮空间。同上。

例：　|　　.BLKW　　n[,value]

```
.BLKW     10
.BLKW     5,1234H
```

RAM_DS　　把当前的 RAM 地址或寄存器地址分配给相应的变量。

```
.RAM_ORG    40H
A:  .RAM_DS     1           ;40H
B   .RAM_DS     1           ;41H
E   .RAM_DS     2           ;42H
C   .RAM_DS     1           ;44H
D   .RAM_DS     1           ;45H
    .ORG        20H
    LD      A,B             ;LD   40H,41H
    LD      C,D             ;LD   44H,45H
```

TACLL,TJP　　只适应于 SAM4

.MACRO　　宏指令。在定义宏体参数时，各参数用‘，’隔开。利用符号传递参数，而不是具体的值。在调用宏的时候，必须先定义，而宏的编译是在调用的时候编译的，不是在定义的时候编译的。

格式：　　||　　.MACRO　　f,……..

```
……
.ENDM
```

定义宏的时候，必须有标号。关于局部标号/全局标号，在三星编译器中，不区分局部标号和全局标号。

```
RAMCLR:  .MACRO   S1,S2
         LD       R0,S1
         LD       R1,S2
CLR:     LD       R0,#0
         INC      R0
         CP       R1,R0
         JR       NE,CLR
         .ENDM
……….
RAMCLR   #00H,#80H
```

条件编译伪指令

IF/ELSE/ENDIF　　条件编译伪指令将使编译器根据前面的定义条件来禁止/允许编程者编写的部分源代码。

格式：　　|　　.IF　　n

```
…….          ;如果 n 为真
```

```
                .ENDIF
或      |       .IF    n
                …….             ;如果 n 为真
                .ELSE
                ……              ;如果 n 为假
                .ENDIF
例：            .IF     VALUE>40
                 ADD    R0,R1    ;如果 n 为真
                .ELSE
                 ADD    R3,R4    ;如果 n 为假
                .ENDIF
```

.IFDEF/ELSE/ENDIF　　IFNDEF/ELSE/ENDIF

根据标号是否在前面定义来决定是否编译编程者编写的部分源代码。

```
格式：      |       .IFDEF/.IFNDEF      |
                ……….    ；标号如果定义，则编译/标号没有定义，则编译
                .ENDIF
或      |       .IFDEF/.IFNDEF      |
                ……..
                .ELSE
                …….
                .ENDIF
```

符号的定义用.EXTERN 或.DEFVAR 伪指令，不可用.GLOBAL 伪指令

INCLUDE　　INCLUDE 可以加载.EXTERN 文件，宏文件，EQU 列表文件，REG 文件。

RADIX　　设置缺省进制

```
格式：      .RADIX    X
例：        .RADIX    H
             ADD      R0,#10      ;16 进制
            .RADIX    DEC
             ADD      R0,#10      ;10 进制
```

.END　　结束程序体。在 INCLUDE 文件中，不要用 END 伪指令。

.算术操作运算

操作符号	单目/双目	优先级	描　述	操作符号	单目/双目	优先级	描　述
(	--	1 最高		<<	双目	4	左　移
)	--	1 最高		>>	双目	4	右　移
+	单目	1	正	&	双目	5	与(3)
-	单目	1	负	|	双目	5	或(4)
>	单目	1	低字节(1)	^	双目	5	异　或
<	单目	1	高字节(2)	&&	双目	5	逻辑与
~	单目	1	取　反	||	双目	6	逻辑或
*	双目	2	乘　法	= =	双目	6	相　等
/	双目	2	除　法	！=	双目	6	不　等
%	双目	2	取　模	>	双目	6	大　于
+	双目	3	加　法	<	双目	6	小　于
-	双目	3	减　法	>=	双目	6	大于等于
				<=	双目	6	小于等于

(1)　此单目运算所进行的操作为（num&0FFH）

(2)　此单目运算所进行的操作为（(num&0FF00H）>>8）

(3)　此双目运算所进行的操作为操作数之间按相对应的位进行与运算

(4)　此双目运算所进行的操作为操作数之间按相对应的位进行或运算

例：(1)
```
LD      R0,#1+2*3/4
LD      R1,#0F0H&INPUT
LD      R1,#03H|INPUT
LD      R1,#(>1234H)      ;34H
LD      R1,#(<1234H)      ;12H
```

(2)
```
mins    .EQU    60*24
LD      R1,#1234>>4       ;0123H
```

(3)
```
.IF(VAL==VAL2)&&(VAL2==VAL3)
LD      R3,#45H
.ENDIF
```

(4)
```
.IF(VAL1==VAL2)
    .IF(VAL2==VAL3)
      LD    R3,#45H
    .ENDIF
.ENDIF
```