# 第三章 AVR 单片机开发工具

**说明:** 为了使读者和用户迅速掌握 AVR 指令系统的功能,边学习,边实践,希望大家先学习<<第三章 AVR 开发工具>>。根据我们的实际教学经验,有的书籍是根据英文源文翻译,程序及说明可能不合中国人习惯,又由于印刷等多种原因,内容有出入,学起来较难。我们是参考有关资料,并在实际工作中验证,并编写有关测试程序(含中文注释),在模拟调试软件窗口观察通过,或在实时仿真器或在 SL-AVR 开发下载实验器上验证通过,把测试实验程序刻在光盘上(也可上网下载 http://WWW.SL.COM.CN),保证用户学习、实验时少走弯路。所以我们先学习系统软件的使用,然后学指令系统,用户一边学习 AVR 指令系统, 一边学习系统软件编程调试,这样使指令功能流向看得见听得见,学习起来有声有色,达到事半功倍的效果。当学完所有指令,你也学会了用软件编程开发调试。我们的想法希望你能去边学边实践,并得到你的认可,我们就谢谢了。

AVR 编辑、编译,调试,下载软件升级较快,书本内容永选跟不上技术的发展,你要获得最新软件只有上网下载。

## 3.1 AVR 单片机的编辑、编译

AVR 单片机实用程序源文件供用户学习参考,今后还将不断增加新内容,也欢迎用户来交流新程序。源程序在 :\AVR\AVR\asmpack\appnotes 或 SL-AVR 目录下,源程序经编译(Assembler)后生成.OBJ 调试文件,HEX 下载文件,LIS 列表打印文件。

### 3.1.1 AVR Assembler 编辑、编译文件的安装与打开:

打开光盘文件 *:\AVR\AVR\asmpack 文件夹,双击图标 

安装。

安装好后双击图标 

进入源文件编辑、编译窗口,
也可使该图标 移到桌面成快捷菜单,
点击图标 进入 AVR Assembler 源文件编辑汇编窗口。

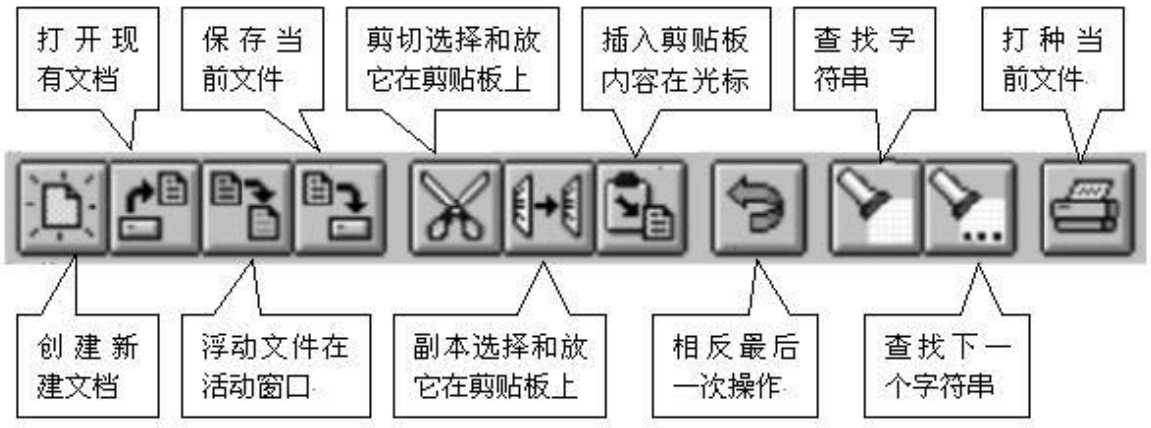


图 3.1 编辑窗口中工具条的快捷按钮(图标)

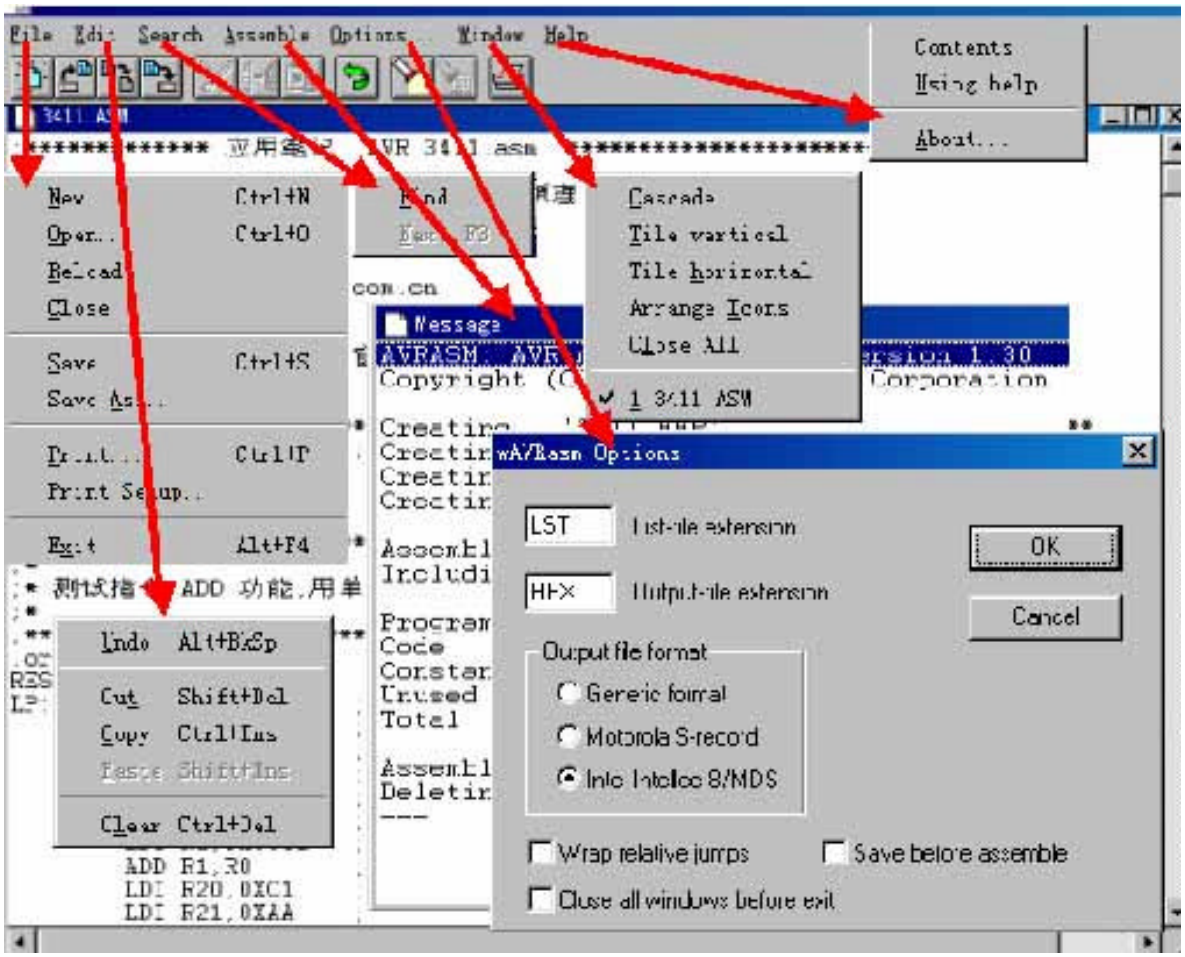


图 3.2 编辑、编译窗口菜单

## 3.1.2 AVR 单片机汇编语言源程序举例:

例一: 文件头不可少,以便了解该程序有关资料

```
;************* 应用举例  AVR 3411.asm  ************************
;*  <4411>表示对应第几章第几节第几段第几个实例
;* 标题:     测试指令功能原理
;* 版本:     1.0
;* 最后更新日期:2000.08.08
;* 支援 E-mail: gzsl@sl.com.cn
;* 描述
;*  用 AVR Studio 调试软件窗口观察指令执行变化情况
;* 作者: SL.
;* 程序适用于所有单片机
;*******************************************************************************
.include "8515def.inc"   ;在编译调试中用到,决不可少" *.inc "文件头
.org  $0000
    rjmpRESET        ;复位

;*******************************************************************************
;*
```

```
;* 测试指令 ADD 功能,用单步或连续单步调试
;*
;*****************************************************************************
.org $0010              ;跳过中断区
RESET:
LP: LDI R16,0X11        ;立即数送寄存器,LDI 指令中寄存器必须 R≥R16,才能汇编成功!
    LDI R17,$33         ;0X11,$33 均为十六进制表示法
    ADD R17,R16         ;R17=0X44 SREG=0X00,H=0,S=0,V=0,N=0,Z=0,C=0,高位低位均无进位
    STS 0X0060,R16      ;内部 SRAM 地址必须≥0X0060,0X0060=0X11
    STS 0X0061,R17      ;0X0011=0X44
    LDS R0,0X0060       ;R0=0X11
    LDS R1,0X0061       ;R1=0X44
    ADD R1,R0           ;R1=0X55  SREG=0X00 , 高位低位均无进位
    LDI R20,0XC1;
    LDI R21,0XAA;
    ADD R21,R20         ;R21=0X6B SREG=0X19,S=1,V=1,C=1 ,高位有进位,低位无进位
    LDI R22,0X46;
    LDI R23,0X6A;
    ADD R23,R22         ;R23=0XB0 SREG=0X2C,H=1,V=1,N=1 , 高位无进位,低位有进位
    LDI R24,0XFF;
    LDI R25,0XFF;
    ADD R25,R24         ;R25=0XFE SREG=0X35,H=1,S=1,N=1,C=1 , 高位有进位,低位有进位
    RJMPLP              ;可循环反复调试检查

                        ;调试时打开 Registers(寄存器窗口),
                        ;Processor(处理器窗口--程序,堆栈,状态寄存器,X/Y/Z 等项),
                        ;New Memory View(存储器窗口--数据,I/O,E2PROM,程序存储器窗口,
                        ;该程序仅需打开片内 SRAM 数据窗口)
```

例二:利用另存文件名,把例一变为例二,仅略修改文件头

```
;************** 应用举例  AVR 4411B.asm  ************************
;* 标题:     测试指令功能原理
;* 版本:     1.0
;* 最后更新日期:2000.08.08
;* 支援 E-mail: gzsl@sl.com.cn
;* 描述
;*  在 AVR Studio 调试软件窗口中置数方法输入数据,用单步观察指令执行变化情况
;* 作者: SL.
;* 程序适用于所有单片机
;*****************************************************************************
.include "8515def.inc" ;在编译调试中用到, 绝不可少" *.inc "文件头
.org  $0000
    rjmpRESET       ;复位
```

```
;*********************************************************************
;*
;* 测试指令 ADD 功能,用置数方法输入数据,用单步或连续单步调试
;*
;*********************************************************************

.org $0010            ;跳过中断处
RESET:
LP:
                ;在寄存器窗口中点击 R16,修改 R16=0X11
                ;在寄存器窗口中点击 R17,修改 R17=0X33
    ADD R17,R16 ;R17=0X44 SREG=0X00,H=0,S=0,V=0,N=0,Z=0,C=0  , 高位低位均无进位
                ;在寄存器窗口中点击 R0,修改 R0=0X11
                ;在寄存器窗口中点击 R1,修改 R1=0X44
    ADD R1,R0   ;R1=0X55  SREG=0X00 ,            高位低位均无进位
                ;在寄存器窗口中修改 R20=0XC1
                ;在寄存器窗口中修改 R21=0XAA
    ADD R21,R20 ;R21=0X6B SREG=0X19,S=1,V=1,C=1  ,高位有进位,低位无进位
                ;在寄存器窗口中修改 R22=0X46
                ;在寄存器窗口中修改 R23=0X6A
    ADD R23,R22 ;R23=0XB0 SREG=0X2C,H=1,V=1,N=1   ,       高位无进位,低位有进位
                ;在寄存器窗口中修改 R24=0XFF
                ;在寄存器窗口中修改 R25=0XFF
    ADD R25,R24 ;R25=0XFE SREG=0X35,H=1,S=1,N=1,C=1  ,     高位有进位,低位有进位
    RJMPLP  ;调试时打开 Registers(寄存器窗口),
                    ;Processor(处理器窗口--程序,堆栈,状态寄存器,X/Y/Z 等项),
                ;New Memory View(存储器窗口--数据,I/O,E2PROM,程序存储器窗口,
                    ;该程序仅需打开片内 SRAM 数据窗口)
```

**编辑源程序文件注意事项:**

程序编译出错,有程序错误定位提示,指示错误原因及错误行号,只需鼠标点击错误提示,光标自动转到源程序的错误行,该行变为红色,以示注意,请修改,见图 3.3。

注意指令中的 Rd,Rr,K,k,P,SRAM 等的数据选择范围;

指令中小写 d 为 16≤d≤31 的指令有:SUBI/SBCI/CPI/ANDI/CBR/ORI/SBR/SER/LDI;

其余指令中 Rd,Rr 的 R 为 0≤d≤31 , 0≤r≤31;

指令中大写 K 为 0≤K≤63 的指令有:ADIW/SBIW;

指令中小写 k 为-64≤k≤+63 的指令有:BRBS/BRBC/BREQ/BRNE/BRCS/BRCC/BRSHBRLO/BRMI BRPL/BRGE/BRLT/BRHS/BRHC/BRTS/BRTCBRVS/BRVC/BRIE/BRID;

指令中小写 k 为 0≤k≤255 的指令有:SUBI/SBCI/CPI/ANDI/CBR/ORI/SBR/LDI;

指令中小写 k 为 -2K≤k≤2K 的指令有:RJMP/RCALL;

指令中小写 k 为 0≤k≤65535 的指令有:LDS/STS;

指令中小写 k 为 0≤k≤4M 的指令有:JMP/CALL;

指令中大写 P 为 0≤P≤31 的指令有:SBI/CBI;

指令中大写 P 为 0≤P≤63 的指令有:IN/OUT;

片内 SRAM 地址 ≥0X0060;


图 3.3 编译出错,错误定位窗口

## 3.1.3 源文件说明:供用户学习 AVR 汇编语言编程时参考

1. avrled.asm 验证 SL-AVR 万用串行下载开发实验器及 AT90S1200 的 A 口、B 口 LED 灯亮灭程序,也同时验证实验器通讯接口连机正常否,avrled2.asm 和 avrled3.asm 仅延时常数不同,所以 LED 闪动快慢不同;
2. AVRSTEP.ASM 用模拟调试单步验证 AT90S1200 B 口、D 口的输出状态;
3. DIP40LED.ASM 验证 SL-AVR 万用串行下载开发实验器及具有 DIP40 封装的 AT90S4414/AT90S8515/AT90S8535 等器件的 A 口、B 口、C 口、D 口 LED 灯亮灭程序,可修改延时常数;
4. AVR100.ASM (访问 E2PROM);
5. AVR102.ASM (数据块传送);
6. AVR108.ASM 加载程序存储器;
7. AVR128.ASM 安装和应用相同比较仪;
8. AVR200.ASM (乘法和除法应用一);
9. AVR200B.ASM (乘法和除法应用二);
10. AVR202.ASM (16 位运算);
11. AVR204.ASM (BCD 运算);
12. AVR220.ASM (冒泡分类算法) ;
13. AVR222.ASM (8 点平均滤波);
14. AVR235.ASM (CRC 程序存储的检查);
15. AVR240.ASM (4X4 键区休眠触发方式);
16. AVR242.ASM (多工法驱动和键区扫描);
17. AVR300.ASM (I2C 总线);
18. AVR302.ASM (I2C 工作);
19. AVR304.ASM (半双工中断方式 UART 应用一);
20. AVR305.ASM (半双工中断方式 UART 应用二);
21. AVR320.ASM (SPI 软件);

22. AVR400.ASM (设置和使用模拟比较器);
23. AVR401.ASM (8 位精度 A/D 转换器);

## 3.1.4 AVR 汇编器

AVR 汇编器覆盖了 AT90S 微控制器家族的全部范围。汇编器用于把汇编代码编译成目标代码，生成的目标代码可以用于模拟仿真器的输入或 ATMEL AVR 在线仿真器的输入。汇编器还能产生能够直接写入程序存储器和 E2PROM 存储的 PROM(可编程只读存储器)代码和一个任意 E2PROM 文件。

汇编器产生固定的代码分配，因此没有链接的必要。

汇编器可在 Miccrosoft Windows 3.11、Microsoft Windows 95/98/2000 和 Microsoft Windows NT 下运行。另外，还有一个 MS—DOS 版本。Windows 版本包括一个在线帮助功能，覆盖了所有这些说明。

### 3.1.4.1 编译器快速启动

AVR 编译器和所有配的程序文件都正确地安装在你的计算机上，请参考 3.1.1 安装与打开。

**一、开始**

开始编辑 AVR 文件。通过从菜单中选择“File→Open”或按下工具条中的图标，打开文件“4411.asm”。这样就把汇编文件装入了编辑窗口，读读程序头，看看程序，但是不要改动它。

**二、编译第一个文件**

一旦看完了这个程序，从菜单中选择编译，第二个窗口（信息窗口）就会出现，并包括一些错误信息。这个窗口将会覆盖编辑窗口，所以在屏幕上应先清除工作空间。选择包含程序代码的编辑窗口，并从菜单中选择“Window→Tile Horizontal”。让编辑窗口比信息窗口大一些是比较好的，所以让信息窗口向下移动一点，并让它挨着编辑窗口的底部，屏幕上将会出现如图 3.3 所示的内容。

**三、寻找和纠正错误**

从信息窗口来看，好像正尽力编译一个有很多缺陷的程序，为了进行下一步，错误必须被发现和纠正。指向信息窗口中的第一个错误，按下鼠标左键。注意，在编辑窗口，一个红色的横条覆盖在行上。错误信息指出 R 必须等于大于 R16。 修改后再编译就 OK。

**四、 重新编译**

为查明是否所有的错误都已改正，双击任意一个错误（为了激活编辑窗口）或在再次编译之前，单击编辑窗口。如果到现在你都这样做了，信息窗口会告诉你已经顺利完成了。

### 3.1.4.2 Microsoft 窗口特性

本节描述 WAVRASM 的特性，仅描述汇编器菜单项的特性。这是假设用户已经对 Windows 菜单项比较熟悉。一个汇编器编辑的典型例子如图 3.3 所示。

**一、打开一个汇编文件**

可以在 WAVRASM 中打开一个新的或已经存在的文件。理论上来说一次打开多少个文件是没有限制的，但是 WAVRASM 有一个限制，就是每个文件的尺寸必须限制在 28K 以下。编辑比这大的汇编文件也是可能的，但是它们不能在一个完整的编辑器中编辑。每打开一个汇编文件，都将产生一个新的编辑窗口。打开二个汇编窗口,对复制程序,程序注释带来方便。

单击工具条中的新建按钮或从菜单中选择“File→New”（Ctrl + N）以创建一个新的汇编文件。单击工具条中的打开按钮或从菜单中选择“ File→Open”（Ctrl + O）以打开一个已经存在

的汇编文件。

## 二、完整的编辑器

当 WAVRASM 装入了一个文件，文本编辑器将被激活。一旦一个文件被装入汇编器的编辑窗口，插入点光标就出现在窗口的左上角。

## 三、输入和格式文本

当输入时，插入点光标向右移动。如果文本输入超过右边边界，文本将自动向左滚动以使插入点光标可见。

## 四、移动插入点

只要把鼠标光标移动到想放插入点的地方并按下左键，插入点光标就可以移动到任何地方。用键或键的组合移动插入点，见表 3.1。

| 移动插入点 | 按键 | 移动插入点 | 按键 |
|---|---|---|---|
| 向文本的右边移动 | 右方向键 | 到一行文本的起点 | Homn 键 |
| 向文本的左边移动 | 左方向键 | 到一行文本的结尾 | End 键 |
| 向文本的上边移动 | 上方向键 | 到文本的起点 | Ctrl+Home 键 |
| 向文本的下边移动 | 下方向键 | 到文本的结尾 | Ctrl+End 键 |

表 3.1 键盘移动插入法

## 五、格式文本

表 3.2 中描述的键是为了得到一定文本形式的必要操作。

表 3.2 用键格式文本

| 操作 | 按键 | 操作 | 按键 |
|---|---|---|---|
| 插入一个空格 | Spacebar | 结束行 | Enter |
| 向右删除一个字符 | Del | 缩排一行 | Tab |
| 向左删除一个字符 | Backspace | 插入一个制表停止位 | Tab |

为了分开一行，可以把插入点移动到要断开的位置，然后按下 Enter 键。为了连接两行，可以把插入点移动到要移动行的开始位置，然后按下 Backspace 键，编辑器就会把这一行连接到前一行上。

## 六、滚动

如果文本的一行要比上一次能够显示的长或者宽，这个文件可以通过滚动条来移动。

## 七、编辑文本

编辑菜单中包含一些功能，能够对编辑工作提供很多帮助。文本可以被删除、移动或复制到新的位置。Undo 命令可以用于取消上一次的编辑操作。文本与别的窗口或应用程序之间的文本传输可以通过剪贴板实现。当文本通 Cut 或 Copy 命令删除或复制，这些文本就放置在剪贴板中。粘贴命令把文本从剪贴板复制到编辑器中。

## 八、选择文本

从 Edit 菜单中选择一个命令去编辑文本之前，被操作的文本必须首先被选择。

用键盘来选择文本：

(1) 用方向键把插入点移动到要选择文本的起始部位。

(2) 按住 Shift 键，直到把插入点移动到要选择文本的结束部位，释放 Shift 键。取消这次选择，按任一个方向键。

用鼠标来选择文本：

(1) 把鼠标光标移动到要选择文本的起始部位；

(2) 按住鼠标左键，直到把插入点移动到要选择文本的结束部位，释放鼠标键。

（3）取消这次选择，按下鼠标左键或任一个方向键。

## 九、替换文本

当文本被选择时，可以通过输入新的文本立即替换被选择的文本。当第一个新的字符被输入时；所选择的文本就被删除了。

要替换文本：（1）选择要被替换的文本；（2）输入新的文本。

要删除文本：（1）选择要被删除的文本；（2）按下 Del 键。

要恢复被删除的文本。在删除文本后，应立即单击工具条中的恢复按钮或从菜单中选择“Edit→Undo”（Alt＋ Backspace）。

## 十、移动文本

要想在编辑器中移动文本，可以通过 Cut 命令把要移动的文本复制到剪贴板中，然后用 Paste 命令把它粘贴到新的位置。

要移动文本：

(1) 选择要移动的文本；

(2) 在工具条中按下剪切按钮或从某单中选择“Edit→Cut”（Shift ＋ Del），文本就被放置在剪贴板中；

(3) 把插入点移动到新的位置；

（4）在工具条中按下复制按钮或从菜单中选择“ Edit→Paste” （Shift＋ Ins）复制文本。

如果一些文本要用到一次以上，不必每次都重新输入它。文本可以用 Copy 命令复制到剪贴板中，然后用 Paste 命令把它粘贴到其它地方。

要复制文本：

(1)选择要复制的文本；

（2）在工具条中按下副本选择按钮或从菜单中选择“ Edit→Copy”（ Ctrl＋ Ins），文本就被放置在剪贴板中；

（3）把插入点移动到要放置文本的位置；

（4）在工具条中按下复制按钮或从某单中选择“Edit→Paste”（ Shift＋ Ins）复制文本。

## 十一、取消一次编辑保作

Undo 命令可以用于取消上一次的编辑操作。例如，文本可能被意外地删除或复制到一个错误的位置。如果在错误发生后立即选择 Undo 命令，文本将被恢复到错误发生以前的状态。为了取消上一次的编辑操作，在工具条中按下恢复按钮或从菜单中选择“Edit→Und。”（Alt＋Backspace）。

## 十二、单击错误信息

汇编器有一个单击错误信息的功能。当编译完一个程序时，一个信息窗口将出现在屏幕上。如果有错误出现，这些错误将排列在信息窗口中。如果信息窗口中的一个错误信息被单击，相应的源代码行就变成了红色，见 3.3 图。如果错误信息出现在包含文件中，什么也不会发生。

如果信息窗口行被双击,包含错误信息的窗口将变成活动窗口,光标将放置在包含错误信息行的起始部位。如果包含错误信息的文件未被打开，这个文件将被自动打开。注意这个功能仅对编译过的文件有效。这就是说，如果源代码文件的行被增加或重新移动过，这个文件必须被重新编译，以得到正确的行数。

## 十三、设置程序选项

WAVRASM 的一些缺省值可以在选项菜单中被修改。如果在某单中选择 Option，图 3.4 所示的对话框将被弹出。在标有“ List— file extension”的框中，缺省的列表文件扩展名被填写。在标有“ Output —file-extension”的框中，缺省的输出文件扩展名“HEX”被填写。在标有“Output file format”的框中，输出文件的格式可以被选择。如果单击了 OK 按钮，这些值将在以后汇编器运行时出现。注意目标文件（用于模拟仿真）将不被这些选项影响。目标文件的扩展名总是 OBJ，格式也是相同的。如果在源代码中定义了 E2PROM 段，汇编器将产生一个以 EEP 为扩展名的文件。这个文件

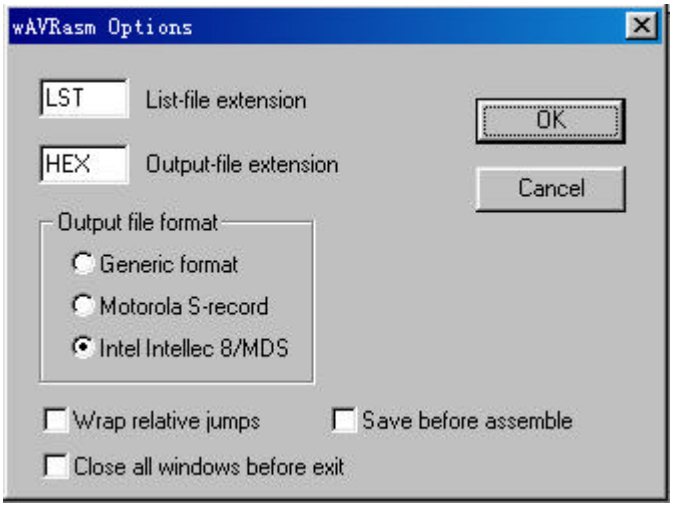


图 3.4 汇编器程序选项

用于初始化 E2PROM 存储器的值。E2PROM 初始化文件的格式与被选择的输出文件的格式相同。“Wrap relative jumps”选项告诉汇编器使用地址约束方式。这个特性仅用于有 4K 程序存储器的器件的编译。在那样的器件上用此选项，相应的跳转指令和调用指令将能到达所有的程序空间。“Save before assemble”选项使汇编器在编译之前自动地保存编辑器中的内容。

## 十四、命令行方案

在 MS—DOS 命令行方案中，汇编器可以通过命令调用。

AVRASM [—m | —I | —g] [—w] input.asm output.lst output. rom

AVRASM 将从 input.asm 中读入源代码，产生列表文件 output.lst、output.asm 和目标文件 input.obj。目标文件将被 MS—Windows 模拟仿真器调用。

用户可以通过选项 -m（Motolora S-record）、-I(Intel Hex)、-g（Generic）中的一个选择所产生的输出文件的格式。缺省值是 Generic 文件格式。

—w 选项告诉汇编器使用地址约束方式。这个特性仅用于有 4K 程序存储器的器件的编译。在那些器件上用此选项，相应的跳转指令和调用指令将能到达所有的程序空间。

# 3.2 模拟调试窗口

AVR Studio 是 AVR 微处理器的开发调试工具。AVR Studio 允许用户在 AVR 在线仿真器或内建 AVR 指令集模拟器上（软件模拟仿真）控制程序的运行。 AVR Studio 支持为 AVR 微控制器的 AVR Assembler 编译器生成的*.OBJ 或*.HEX 文件和 IAR C 编译器编译的源代码层次的执行。

AVR Studio 在 Microsoft Windows 95/98/2000 和 Microsoft Windows NT 上运行。

## 3.2.1 安装 AVR Studio 调试工具

打开光盘，*:\avr\avr\Studio 3.X,双击安装图标,安装结束，


双击调试工作图标也可把该图标双击桌面调试工作图标进


进入调试窗口，移到桌面成快捷菜单方式，入 AVR Studio 调试窗口,如图 3.5。

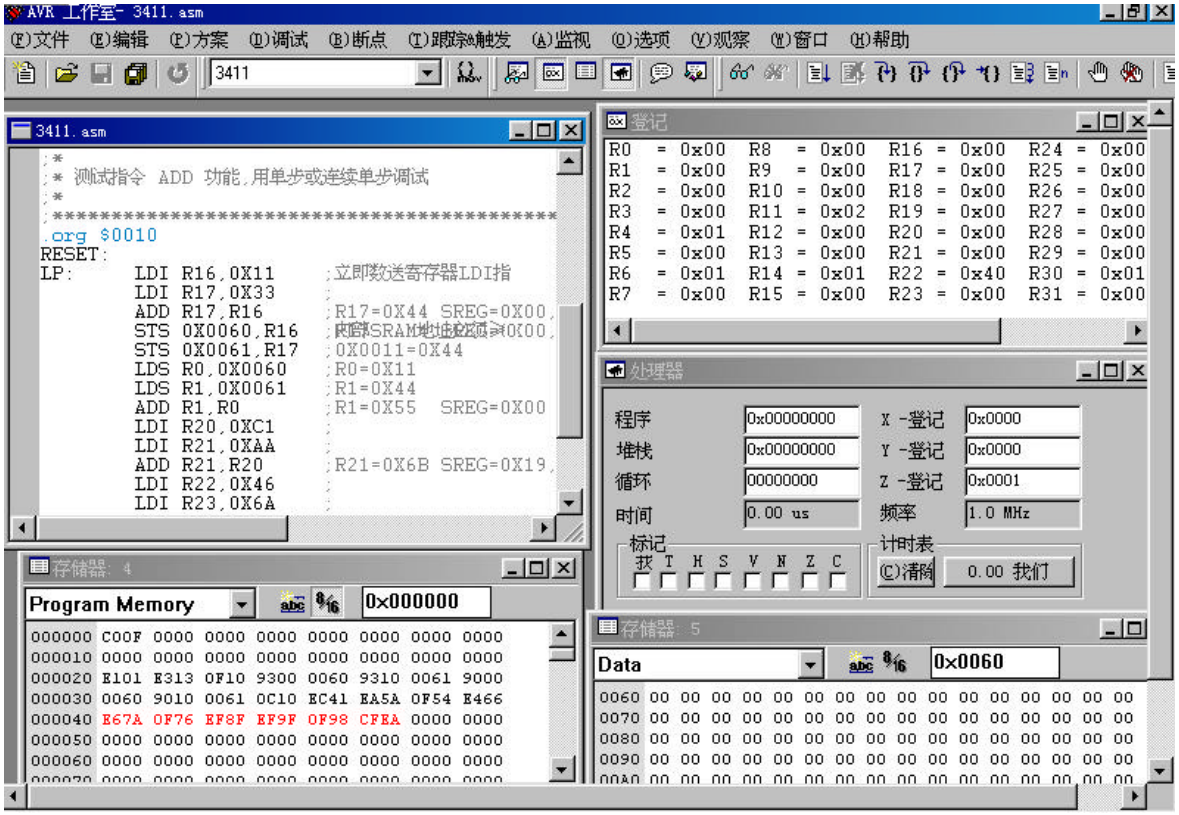


图 3.5 AVR Studio 调试窗口

## 3.2.2 设置器件配置文件 *.aio 方法:

在 AVR Studio 窗口下,打开了顶项菜单,Options ⟶ Simulator Options 出现复选框,在 Device 栏下选择相应器件名,单击 OK 键确认。

然后再打开菜单 View ⟶ New IO View 出现器件配置窗口,双击左侧出现对应观察窗,程序调试时对应参数会变化。尤其对模拟调试观察器件引脚的输出状态,直观明了。


图 3.6 器件配置窗口

## 3.2.3 AVR Studio 调试窗口下拉菜单

一旦安装了 AVR Studio 调试软件，就能够通过双击 AVR Studio 的图标运行它。如果一个仿真器是期望的执行对象，记住在运行 AVR Studio 之前连接好 AVR 在线仿真器。

下面将对 AVR Studio 主要特性进行简要的描述。 AVR Studio 允许在 AVR 在线仿真器或内建 AVR 指令集模拟器上运行 AVR 程序。用 AVR Studio 运行程序，必须首先用 IAR 系统的 C 编译器或用 ATEML 的 AVR 汇编器生成一个能被 AVR Studio 识别的目标文件。

AVR Studio 在执行一个程序时的状态如图 3.5 所示。另外相对于源程序窗口， AVR Studio 定义了一些窗口用于观察微控制器的不同源程序。

版本更高的 AVR Studio 调试窗口,增加了直接进入在线仿真器相关操作及 Tools 选择。


图 3.7 AVR Studio3.0 主菜单工具条窗口


图 3.8 AVR Studio3.2 主菜单工具条窗口

在 AVR Studio 中，最关键的窗口是源程序窗口。当打开目标文件时，源程序窗口也就自动产生了。源程序窗口显示在执行对象（如仿真器或模拟仿真）中，当前正被执行的代码。文本指示总是放在下一条将被执行的语句上，状态条指出执行目标是 AVR 在线仿真器还是内建指令集模拟

器。

缺省时，假设是在源代码层次上执行。所以，如果源程序存在，程序将在源代码层次模式启动。另外，相对于 C 和汇编程序的源代码(*.OBJ)层次执行，AVR Studio 也能够在(*.HEX)反汇编层次上执行程序。当一个执行的程序被停止后，用户可以在源模式和反汇编模式下互相转换。

在 AVR Studio 中所有必要的执行命令都是有效的，用户可以通过跟踪执行、单步执行功能，把光标放在一条语句上，直到执行到那条语句，停止执行来执行程序。另外用户可以有不限数量的代码断点，每个断点都能定义为可用或不可用。断点在对话中被装入。

源程序窗口结出关于程序控制结果的信息。另外，AVR Studio。提供一些其它的窗口，便于用户对执行目标的每个元素的状态都能完全控制。有如下可用的窗口。

(1) 监视窗口：显示定义符号的值。在监视窗口中，用户可以监视如 C 程序中变量的值。

（2）寄存器窗口：显示寄存器文档的内容。当执行停止后，可用于修改寄存器的内容。

（3）存储器窗口：显示程序存储器的内容。如数据存储器、I／O 存储器或 E2PROM 存储器，程序存储器,这些存储器可用十六进制值或 ASCII 码观察。当执行停止后，存储器的内容都能被修改。

（4）外设窗口：显示与不同外设相联系的状态寄存器的内容，如 E2PROM 寄存器、I／O 端口、定时器等。

（5）信息窗口：显示 AVR 给用户的信息。

（6）处理器窗口：显示执行目标的重要信息。包括程序计数器、堆栈指针状态寄存器、时钟周期数。当执行停止后，这些单元可以被修改。

当首次打开一个目标文件，用户需要设置一些便于程序观察的窗口，由此把屏幕上的信息做成特殊的工程文件。当下次这个目标文件被调入时，设置就自动的恢复了。

## 3．2 .4　AVR Studio 窗口

### 一、源程序窗口

在 AVR Studio 对话框中源程序窗口是一个重要的窗口。当打开一个目标文件时它就被创建了，并一直贯穿于整个过程。如果关掉源程序窗口，对话也就被中断。

源程序窗口显示当前正被执行的代码。图 3.9 就是一个源窗口的例子。

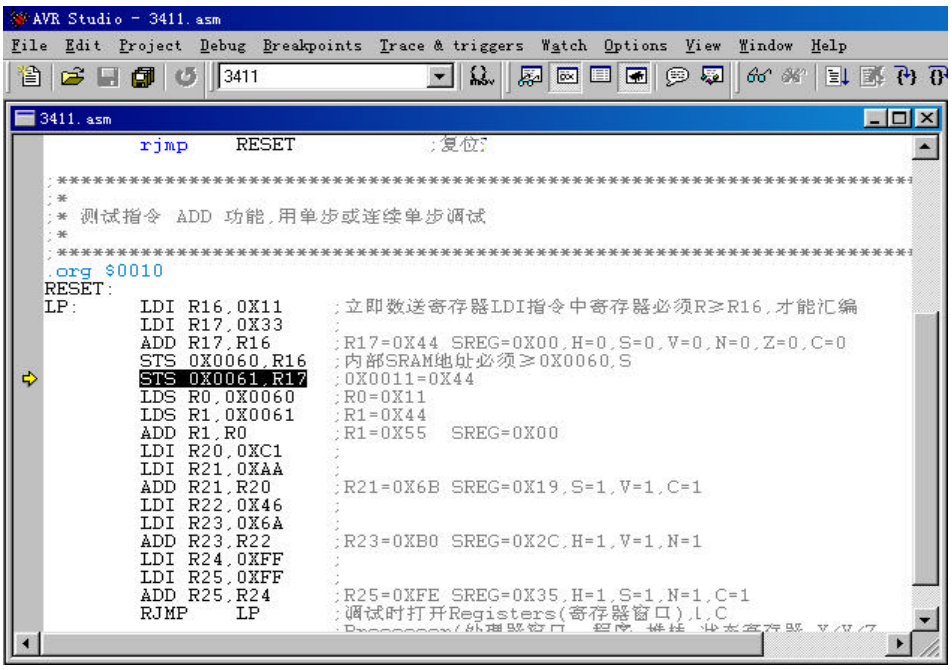


图 3.9

下一条将被执行的指令总是被 AVR Studio 标注着。如果标注被用户移动，在先前被标注的文本变成红色之后，这下一条语句仍被识别。

源程序窗口中，在语句的左边有断点的地方用一个圆点作为断点的标识。

如果按下模式选择框的右边按键，源窗口将在源代码层次和反汇编层次执行上互相转换。当 AVR Studio 在反汇编模式时，所有的操作，例如单步执行将在反汇编级别上执行。如在一些情况中，没有源代码层次的信息可用，（如，把一个 Intel 十六进制文件作为目标文件，没有源代码层次信息可用时），执行只能在反汇编层次上进行。

触发断点，运行到光标处和复制功能，也可以通过在源程序窗口中按下鼠标右键来实现。当鼠标右键被按下，一个菜单就出现在屏幕上，如图 3.10 所示。

如果光标放在一条指令上，一个运行到光标处的命令被给出，则程序将一直执行到光标放置的那条指令上。以同样的方式可以设置断点：把光标放在一条语句上，给出一个 Toggle Breakpoint 的命令。如果断点已经设置在这条语句上，则断点将被解除。如果没有断点设置在这条语句上，则就产生一个断点。

一个目标文件可以由几个模块构成，在同一时刻只能显示一个模块，但是用户可以通过选择在源程序窗口左上方的选择框转换到别的模块。这是一个有用的特性，可以在一个当前模块激活时，在其它模块观察和设置断点。源程序窗口支持 Windows 剪贴板。用户可以选择源程序窗口中全部或一部分内容，然后通过从编辑菜单中选择复制的方法把它复制到剪贴板上。

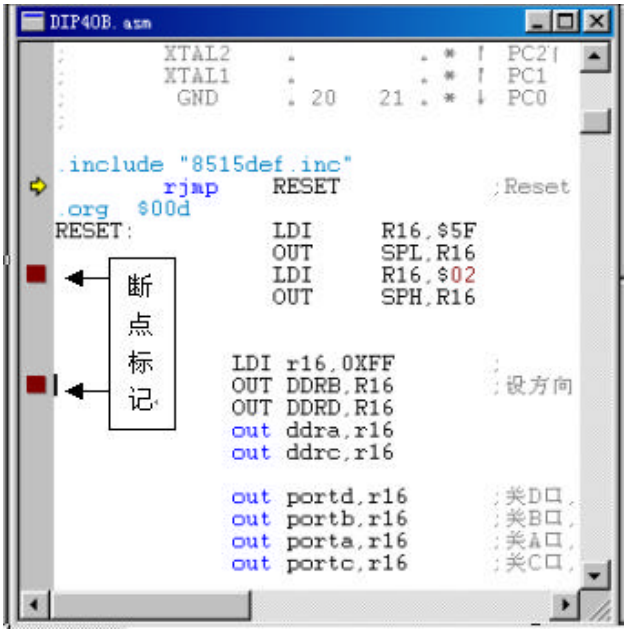


图 3. 10 设置断点

## 二、监视窗口

监视窗口可以显示像 C 程序中变量一样符号的类型和值。因为 AVR 汇编器不会产生任何符号信息，这个窗口只能在执行 C 程序时有意义。图 3. 11 给出一个监视窗口的例子。

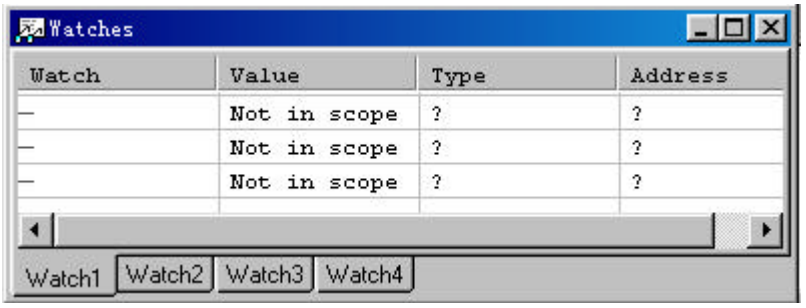


图 3. 11

这个监视窗口有三部分，第一部分是被监视符号的名字，第二部分是符号的类型，第三部分是符号的值。监视窗口在缺省状态下是空的，即用户所有想监视的变量必须被加到监视窗口中。一旦一个符号加了进去，在下次程序执行时它就会重新出现。当监视窗口关闭时，这些加入的监视量也被保存。这里有增加监视量、删除监视量和删除所有监视量的命令。一个监视量的加入可

以通过从调试条或监视菜单中给出一个增加监视量命令。如果监视窗口是活动窗口，也可以通过按下 Ins 键给出一个增加监视量命令。增加监视命令给出后，用户必须键入符号的名字。用户也可以输入一个带有或没有范围信息的符号名字。

AVR Studio 先以符号包含范围信息来搜寻符号。如果那样的符号没有找到，AVR Studio 把符号的名字放在当前的范围上，然后搜寻这个新符号。如果仍没有这样的符号被发现，这个符号就被解除了。在类型部分出现“？？？”，值的部分一直空着。如果找到了这个符号名称，这个符号就被限制，带有范围信息的符号就显示在监视块，类型和值域也被填充。每当程序执行停止时,AVR Studio 都试着用当前的范围来赋值无限制的符号。

浮点符号是不可用的。一旦符号被限制，它将保持赋值。这些监视量在对话中被保存。不论符号是否被赋值都是这些信息的一部分。如果程序进入一个范围，那里一个被赋值的符号是无效的，值域将变成"out of scope"。

为了删除一个监视量，符号名称必须首先用鼠标左键点上。当用这种方法标注了一个符号，AVR Studio 接收监视菜单中的删除监视量命令。如果监视窗口是当前活动窗口，标注的符号也可以通过 Del 键进行删除。

监视窗口可以用于像监视单个变量一样监视 C 数组和结构体。语法同 C 语言(用〔 〕定义数组，用“.”定义结构体)。没有提及的指针不被支持，当监视数组时，变量可以用于动态地索引数组。例如，可以监视“my_array〔i〕”，如果 i 是一个与数组 my_array 同样范围的整型变量。

同一时刻只能有一个激活的监视窗口，监视符号在对话窗口中恢复，监视窗量也同样地被恢复和保存。

## 三、寄存器窗口

寄存器窗口显示 AVR 寄存器文件中的 3 2 个寄存器的内容。图 3.12 所示是一个寄存器窗口的例子。

当寄存器窗口大小变化时，里面的内容会为更好地适应窗口形状而重新组织。当执行停止后，寄存器窗口里的值可以被改变。为了改变寄存器的内容，首先确定执行是停止的，然后把光标放置在要改变的寄存器上，按鼠标左键两下（不是双击，要在两次击打之间有一个停顿），寄存器就可以被改变了，以十六进制的形式键入新的内容,这对调试程序的参数很有用。最后，按下回车键确认或 ESC 键撤消改变。

同一时刻只能有一个寄存器窗口被激活。

## 四、信息商口

信息窗口显示 AVR Studio 给用户的信息。当一个复位命令执行后，信息窗口的内容被消除。图 7.13 是一个信息窗口的例子。

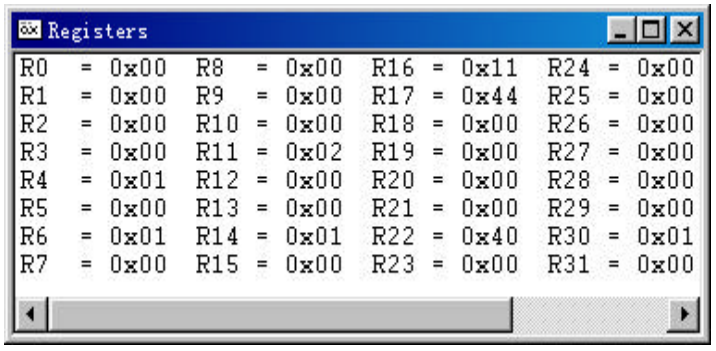


图 3.12

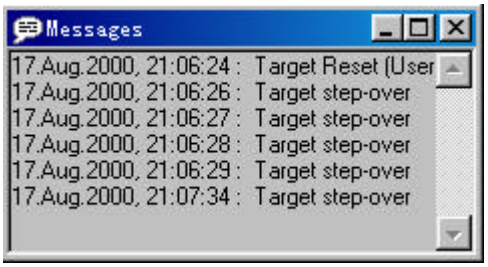


图 3.13

当信息窗口被关闭和再打开时，里面的内容也被恢复，在同一时刻也只能有一个被激活的信息窗口。

## 五、存储器窗口

存储器窗口允许用户观察和修改当前执行对象的不同存储器的内容。同样的窗口可以用来观察所有的存储器形式，存储器窗口可以用来观察数据存储器、程序存储器、I / O 存储器和 E2PROM 存储器。

用户可以有几个共存的存储器窗口。图 3.14 是一个存储器窗口的例子。

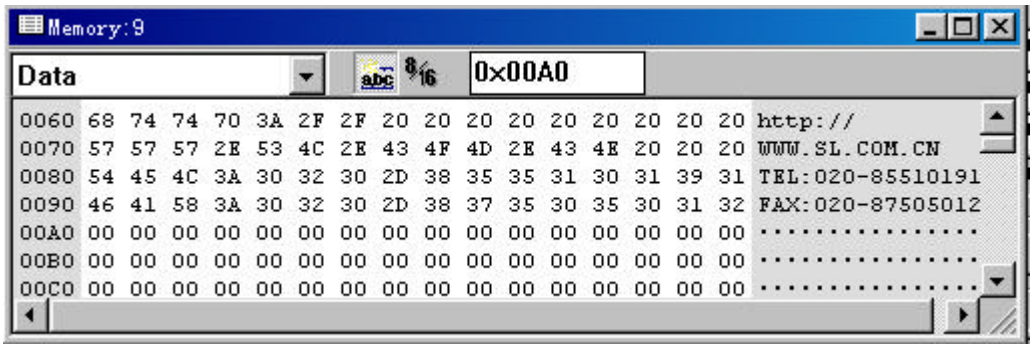


图 3.14

想看哪一种存储器形式，可在存储器窗口左上角的存储器选择框中进行选择。当打开一个新的存储器窗口，数据存储器是缺省的存储形式。

十六进制表示的存储器地址和内容总是显示着的。另外，用户还可以看到存储器内容的 ASCII 码表示。用户也可以选择把十六进制值组织成十六位组或八位组。

当看程序存储器时，在地址列中显示的是字地址。在数据列中，MSB 列在 LSB 列前。

用户可以通过在包含被修改项的行上双击来修改存储器内容。当存储器上的一行被双击，一个窗口出现在屏幕上。如果存储器以八位组观察，修改也是以八位组操作；如果以十六位组观察，则修改也是以十六位组操作。

当以八位组操作，就会出现如图 3.14 所示的窗口。

当，以十六位组操作，就会出现如图 3.15 所示的窗口。

在两种情况下操作是相同的。如果按下 Cancel 键，不会有任何改变；如果按下 OK 键，存储器内容就会根据改变而更新。

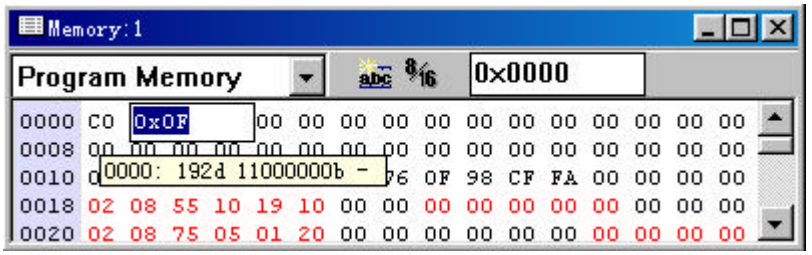


图 3.14 存储器以八位组观察

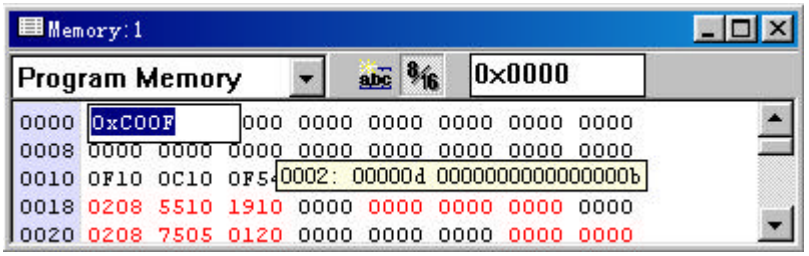


图 3.15 存储器以十六位组观察

## 六、处理器窗口

处理器窗口包含执行对象的重要信息。图 3.16 是一个处理器窗口的例子。

程序计数器指出下一条将被执行指令的地址，且以十六进制形式显示，当执行停止后也可以被改变。当前的指令在程序计数器改变时被取消了。程序计数器改变后，用户必须执行单步功能跳到新的地址上。堆栈指针装着放在 I/O 区的堆栈指针的当前值。如果执行对象有一个硬堆栈代替一个基于 SRAM 的堆栈，就会在堆线指针域中指出。当执行停止后，堆栈指针值可以被改变。

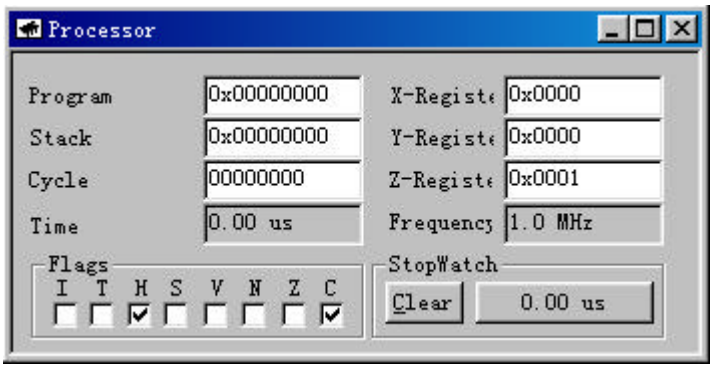


图 3.16 处理器窗口

周期计数器给出自上次复位取消后时钟周期的数据，AVR Studio 在线仿真器运行时不支持周期计数器。因此以仿真器为执行对象时，周期计数器一直为 0，且以十进制显示，在执行停止后也能被改变。

计时栏 StopWatch,在调试程序时很有用,可测某段程序执行多少时间,测延时子程序时间,计算脉冲周期、频率等。

标志位显示当前状态寄存器的值。当执行停止时，这些位可以通过在这些标志上单击来改变。复选的标志指出这个标志被置位（在状态寄存器中相应位是 1）。

在同一时刻只能有一个处理器窗口被激活。

## 七、外设窗口

用户可以在存储器窗口监视 I/O 的内容，但以连续存储结构的方式查看 I/O 区域，并不是方便地观察众多 I/O 设备状态的方法。特殊的外设窗口更具体，使观察 I/O 设备变得简单。

(1) 8 位定时器/计数器窗口（或 16 位定时器/计数器）：8 位定时器/计数器窗口显示 8 位定时器/计数器 0 的所有重要信息。当从快捷键菜单 中选择 8 位定时器/计数器时，定时器/计数器窗口出现在屏幕上，如图 3.17 所示。定时器/计数器框给出 8 位定时器/计数器的值，初值框给人相应的初值，0 通常指出定时器/计数器是关闭的。初值必须在 0~7 之间选择。溢出标志控制框在溢出标志位为 1 时复选，在溢出标志位为 0 时为空。溢出中断框根据允许或屏蔽溢出中断而复选或空。当执行停止后，所有的值都可以被用户改变。

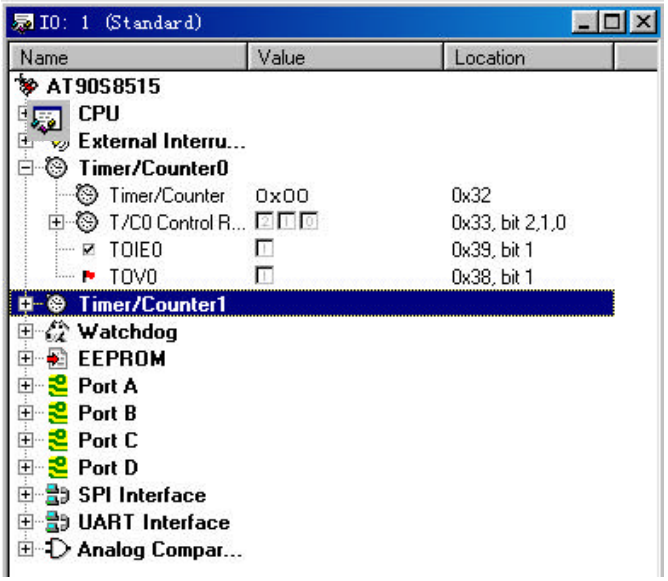


图 3.17 外设窗口

（2）端口窗口：端口窗口显示通常与一个端口联系的三个不同 I／O 寄存器。相应的端口窗口就会出现如图 3.17 所示。端口窗口以十六进制值和一位值显示 I／O 区域的端口的设置，引脚和直接数据寄存器的值。当执行停止后，寄存器的值都可以被改变。

(3)E2PROM 寄存器，图 3.17 所示的窗口就会出现。 AVR Studio 知道在执行对象上有多少 E2PROM 存储器可用，因此如果需要，地址框中包含地址的高位和低位。

### 3.2.5 AVR Studio 命令

AVR Studio 使用了一定数量的命令。命令可用通过不同的方式结出：菜单选择、工具条按钮和热键。

**一、管理**

(1)打开文件 当从 File 菜单中选择了 Open，一个文件选择对话框就会出现在屏幕上（注意 AVR 设定文件扩展名为 OBJ，因此在缺省状态下，仅是有 OBJ 扩展名的文件被列出）。用户必须选择目标文件去执行，AVR Studio 支持下面的格式：

- IAR UBROF
- 通过 ATEML AVR 汇编器生成的 AVR 目标文件
- Intel—Hex

AVR Studio 自动识别目标文件格式。四个最近使用的文件可以在 File 菜单中直接被选择装入。

当打开文件时， AVR Studio 寻找具有相同名字但扩展名为 AVD 的文件。它包含关于工程的信息，包括窗口的位置。如果 AVD 工程文件没有找到，那就仅创建一个源程序窗口。

AVD 文件也包含关于断点的信息。在上次对话中定义的断点重新被装入。如果目标文件比工程文件新，断点就被忽略了。

如果源代码层次信息可用，程序将执行到第一条源语句。

（2）关闭文件 当从 File 菜单中选择 Close，对话中所有的窗口都被关闭。AVR Studio 还在同目标文件一样的目录下写一个文件，包含项目信息。这个文件与目标文件的名称相同，但扩展名为 AVD。

(3)复制文本 用户可以在源程序窗口中标记文本，并通过选择 Edit 菜单中的 Copy 命令把文本复制到 Windows 剪贴板中。

**二、执行控制**

执行控制用于控制程序运行，所有执行命令通过菜单、热键和调试工具条给出都有效。

(1)全速执行 Debug 菜单中的 GO 命令开始（或确定）程序的执行。程序一直执行到被停止（用户操作）或断点出现，GO 命令仅当程序停止时有效。热键为 F5。

(2)运行停止 Debug 菜单中的 Break 命令用于停止程序的执行。当运行停止后，窗口中所有的信息都被更新，Break 命令只在程序运行时有效。热键为 Ctrl＋ F5。

(3)跟踪进入 Debug 菜单中的 Trace into 命令执行一条指令。当 AVR Studio 是在源模式，一条源代码层次的指令被执行。如果在反汇编层次，一条汇编指令被执行。当 Trace into 执行后，所有窗口中的信息都被更新。热键为 F11。

（4）单步执行 Debug 菜单中的 Step over 命令执行一条指令。如果指令包含一个功能／子程序调用，功能／子程序调用也被当一步执行(可称宏单步)。如果在 Step over 中有用户断点，执行就被挂起。当 Step over 执行后，所有的窗口中的信息都被更新。热键为 F10。

（5）单步退出 Debug 菜单中的 Step out 命令执行直到当前功能完成。如果在 Step over 中有用户断点，执行就挂起。如果当程序在顶层时 Step out 命令就发出，程序将一直执行到一个断点或被用户停止。当 Step out 命令完成后，所有窗口中的所有信息都被更新。热键为 Shift

＋F11。

（6）运行到光标处　Debug 菜单中的 Run to Cursor 命令执行，直到程序运行到源程序窗口中光标指出的那条指令。如果在执行 Run to Cursor 命令时有一个用户断点，执行不被挂起。如果光标指出的指令不能到达，程序只能被用户停止。当 Run to Cursor 命令完成后，所有窗口的信息都被更新。热键为 F7。

(7) 复位　Reset 命令完成执行对象的复位功能。如果程序正在执行，当这个命令给出时，执行将被停止。如果用户在源代码层次，程序在复位后，将一直执行到第一条源语句。当复位命令完成后，所有窗口中的信息都被更新。热键为 Shift＋ F5。

## 三、监视量

当在 C 语言源层次上执行，监视窗口可以用来监视符号。当执行由 ATEML AVR 汇编器生成的目标文件时，没有符号信息是可用的，因此监视窗口不能用来监视任何信息。

(1) 添加监视量　为了插入一个新的监视量，用户必须在监视窗口中选择 Add Watch 命令或者在 Debug 工具条中按下 Add Watch 按钮。如果当 Add Watch 命令给出时，监视窗口未被打开，将创建一个监视窗口，已经定义的监视量被插入。如果监视窗口是活动窗口，增加一个新的监视也可以通过按下 Ins 键。

（2）删除监视量　用户可以删除一个监视量。首先在监视窗口选定要删除的符号(通过移动鼠标指针指向监视的名字，按下鼠标左键选定一个监视量)，然后从 Watch 菜单或调试工具条中给出一个 Delete Watch 命令。如果监视窗口是活动窗口，一个选定的符号也可以通过按下 Del 键进行删除。

（3）删除所有的监视量　Watch 菜单中的 Delete a11 watches 命令是有效的。当使用了这个命令，所有定义的监视量都将被从监视窗口中删除。

## 四、断点

用户可以设置不限数量的代码断点。除非产生了一个新的目标文件，否则断点就在对话中恢复。如果目标文件比工程文件新，断点也会被忽略。

当在某处设置了一个断点，在指令左边将有一个圆点指出这是一个断点。

(1) 触发断点　Togglebreak point 命令触发光标处指令的断点状态。注意，这个功能仅当源视图是活动视图时有效。


图 3.18 断点设置列表窗口

（2）清除所有的断点这个功能清除所有定义的断点，包括已经被忽略的断点。图 3.18

（3）查看列表　当 Showlist 被选择,图 3.18 所示的对话框就会出现。在断点对话框中，用户可以观察已经存在的断点，增加一个新的断点，删除一个断点或允许／屏蔽断点。

## 五、工具条

AVR Studio 包含下面描述的三种不同的工具条。工具条可以被单独地插入和移动。如

果需要，也可通过 View→Toolbars 菜单控制或解控。


图 3.19

(1) 通用工具条　包括标准窗口命令的按钮。通用工具条有图 3.19 所示的按钮。

（2）调试工具条　包含执行控制和监视窗口控制的按钮。调试工具条有图　3.19 所示的按钮。

(3)视图工具条　包含允许和屏蔽许多常用窗口和增加存储器窗口的按钮。视图工具条有图 3.19 所示按钮。

## 六、热键摘要

部分热键在 AVR Studio 中的定义见表 3.3。

表 3.3 部分热键定义

| 命令 | 热键 | 命令 | 热键 | 命令 | 热键 |
|---|---|---|---|---|---|
| 改变寄存器窗口 | Alt + 0 | 复制到剪贴板 | Ctrl + C | 运行到光标处 | F7 |
| 改变监视窗口 | Alt + 1 | 打开文件 | Ctrl + O | 改变断点 | F9 |
| 改变信息窗口 | Alt + 2 | 帮助 | F1 | 单步执行 | F10 |
| 改变处理器窗口 | Alt + 3 | 全速执行 | F5 | 跟踪进入 | F11 |
| 增加存储器窗口 | Alt + 4 | 运行停止 | Ctrl + F5 | 单步选出 | Shift + F11 |
| 显示断点列表 | Ctrl + B | 复位 | Shift + F5 | | |

### 3.2.6 执行对象

AVR Studio 可以面向一个 AVR 在线仿真器或内建 AVR 指令集模拟仿真器。当用户打开一个文件，AVR Studio 自动地识别系统的一个串行端口是否有一个仿真器可用。如果发现了一个仿真器，就把它作为执行对象。如果没有仿真器，则执行被 AVR 内建指令集模拟仿真器（软件模拟仿真）代替。状态条指出执行目标是在线仿真器还是内建指令集模拟仿真器。

#### 一、 AVR 在线仿真器

如果系统中 AVR 在统仿真器有效，它就会自动地被作为执行对象。仿真器必须连接在一个串行口上。如果仿真器已与系统相接，但是不能被识别，关掉文件，复位仿真器，再试一次。想了解关于 AVR 在线仿真器的信息，请看 AVR ICE-200 用户手册。

若用户想用模拟仿真器，即使仿真器在系统上，也可用在打开文件前断开或关掉仿真器。用户可以选择仿真器是用板上的可编程时钟，还是用外部时钟(请看在线仿真器手册以获取更多的信息)。如果内部晶振作为时钟来源，用户可以选择从 400~20MHz 的频率(根据器件允许的频率选择)。用户可以选择列表中的典型频率，也可以在信息窗口自定义一个频率图 3.20。在对话框中仿真器的速度被恢复。

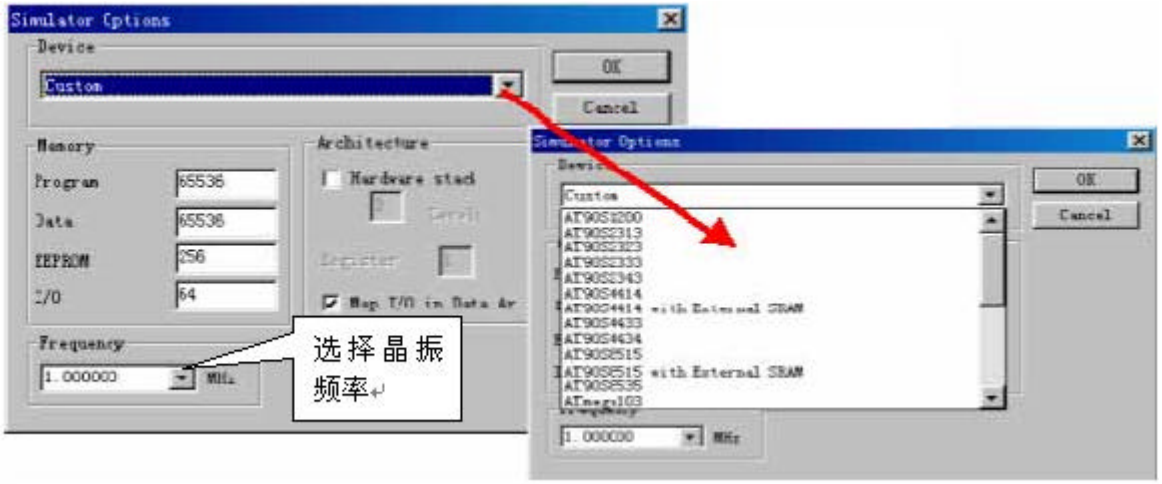


图 3.20 选择器件和频率窗口

#### 二、 AVR 指令模拟仿真器

如果 AVR Studio 不能成功地证明存在一个仿真器，它就用一个内建模拟仿真器代替。模拟仿真器也支持一些 I/O 设备，有不少不同的选项。如果从 Option 菜单中选择 Simulator Options，

图 3.21 所示的对话框就会出现在屏幕上。

在对话框中模拟仿真器选项也被恢复。对话框中所有的数据都是十进制值，改变这些值的任何一个都会迫使AVR Studio执行一个复位命令。

(1) 设备选择　在设备框中，用户可以在多种不同的标准配置中选择。如果选择了一个标准配置，存储器配置和结构细节就相应地填写了，用户也可以定制配置。

（2）定制选择　如果定制选择按钮被按下，则允许用户输入存储器和结构栏的值。

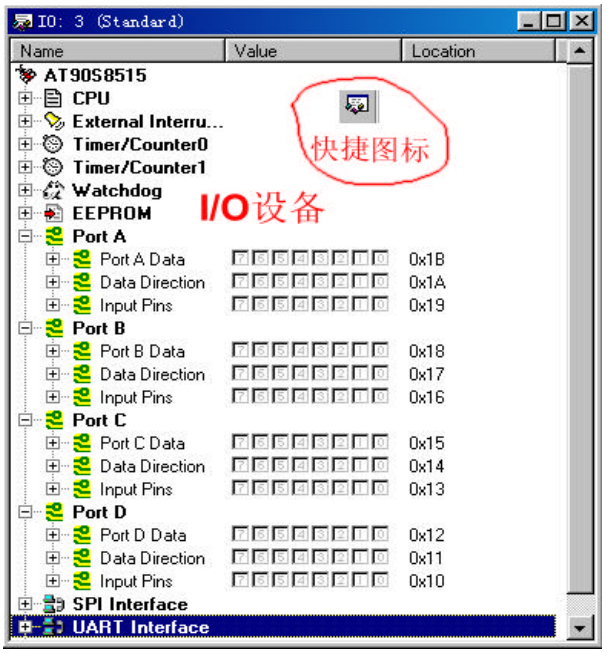


图 3.21 I/O 设备窗口

▪ 程序存储器大小以字计算。可以键入的最大值为 65536。超出这个选择范围，操作是不确定的。

▪ 数据存储器以字节计算。可以键入的最大值为 65536（64K 字节）。注意指针寄存器不是循环的。如果用户使用 SRAM 超过选择的范围，操作是不确定的。在数据存储器栏中的值是能用于寻址 SRAM 的最高地址。寄存器文档的大小和 I／O 区（如果映射在 SRAM 区）必须加到 SRAM 的值上，以得到这个栏的正确值。

▪ E2PROM 的大小以字节计算。可键入的最大值为 65536（64K 字节）。E2PROM 地址寄存器仅仅包括能够寻址 E2PROM 的必要多的位。

▪ I/O 地址大小以字节计算。I/O 地址大小允许的值为 64，128，256。

▪ 用户可以选择模拟仿真器是否使用硬难栈。若选择了硬堆栈，用户可以设置硬堆栈的数目。

▪ 用户不能让模拟仿真器管理几个寄存器文档。

▪ 用户可以选择 I／O 区域是否占用 SRAM 的地址空间。如果 I／O 可以在 SRAM 区寻址，它将在地址 0x20 以上使用。

(2) 记录端口　用户可以记录输出

口的活动。如果从选项某单中选择了 I/O 快捷图标,I/O 对话框就会出现在屏幕上。

用户必须选择的那个端口将被记录。如果选择了一个端口，用户还必须选择一个文件用于放置记录数据。文件上的内容是端口寄存器的内容。记录文件的每行都有下面的格式，周期:数据。周期栏中以十进制格式，数据栏中以十六进制格式。如果在一个周期中端口寄存器的内容没有改变，就没有输出产生。记录文件在每次程序复位时被删除。在每次程序装入 AVR Studio 时，记录都必须被人工激活。

## 三、外部激励

用户可以设置端口的值。

如果选择了一个端口，用户必须指出激励文件的位置。激励文件中的值将在指定的周期放在指定端口的 PIN 寄存器中。激励文件的格式与端口记录文件格式相同。注意，仅当引脚设为输入时有效。

## 四、定时器／计数器 0

模拟仿真器支持定时器／计数器 0。如果选择了一个标准设备，定时器／计数器 0 的溢出中断向量和外围计数器引脚被相应地设置。如果选择了定制外围设备，定时器／计数器 0 的溢出中断向量和外围计数器引脚同 AT90S1200 设置。

## 五、外部中断

模拟仿真器支持外部中断。如果选择了一个标准外围设备，外部中断就被相应地设置了。如果选择了一个定制设备，对于 AT90S1200 将有一个外部中断可用。

# 3.3　AVR 单片机开发下载实验器 SL-AVR

AVR 编程有二种方法:用万用编程器并行编程,适合大批量生产;利用 ISP 信号线实现串行在线下载编程,这是今后 IC 器件发展必然趋势,不需拆下器件编程,对产品升级带来方便,更适合远距离对设备监控维护。

为配合<<AVR 高速嵌入式单片机原理与应用>>一书的出版,在 ATMEL 北京与香港办事处及华东师范大学电子科学技术系 ATMEL 实验室的大力协助下,我们顺利设计完 AVR 单片机开发串行下载实验器 SL-AVR(简称开发实验器)。该开发实验器采用双龙电子的专利技术(专利号:98226094.6),是为 ATMEL 的 AVR 单片机特别研制的单 5V 串行下载、开发(软件摸拟仿真)、实验工具。该开发实验器适用于 ATMEL 公司所有具有串行下载功能的 AVR 单片机,同时还可做 AVR 单片机的 I/O 口、A/D、D/A、音频输出,键盘、LED 数码管、16X2 LCD 液晶显示器、步进马达等实验。该开发实验器提供了 ATMEL 的集成模拟仿真调试软件,对初学 AVR 单片机的设计者,可暂时节省购买较昂贵的实时仿真器及万用编程器的费用。SL-AVR 开发实验器实物图如图 3.22:

## 3.3.1　SL-AVR 开发下载实验器硬件结构

AVR 单片机开发下载实验器 SL-AVR (等于 AVR 编程器+模拟仿真器+实验器), SL-AVR 开发实验器硬件采用模块化设计,便于用户灵活组成你的科研项目所需的硬件结构。硬件有 RS232 通信接口;串行下载监控;DIP8-DIP40 通用锁紧插座,DIP40 端口用短路块连接作输出,用 LED 发光二极管显示器件引脚高低电平,也可用短路块断开,作输入或其它用途;有 6 位 LED 数码管作显示;有 2X16 点阵 LCD 液晶显示器;有 17 键的键盘;有网络电阻作高精度 A/D 转换;

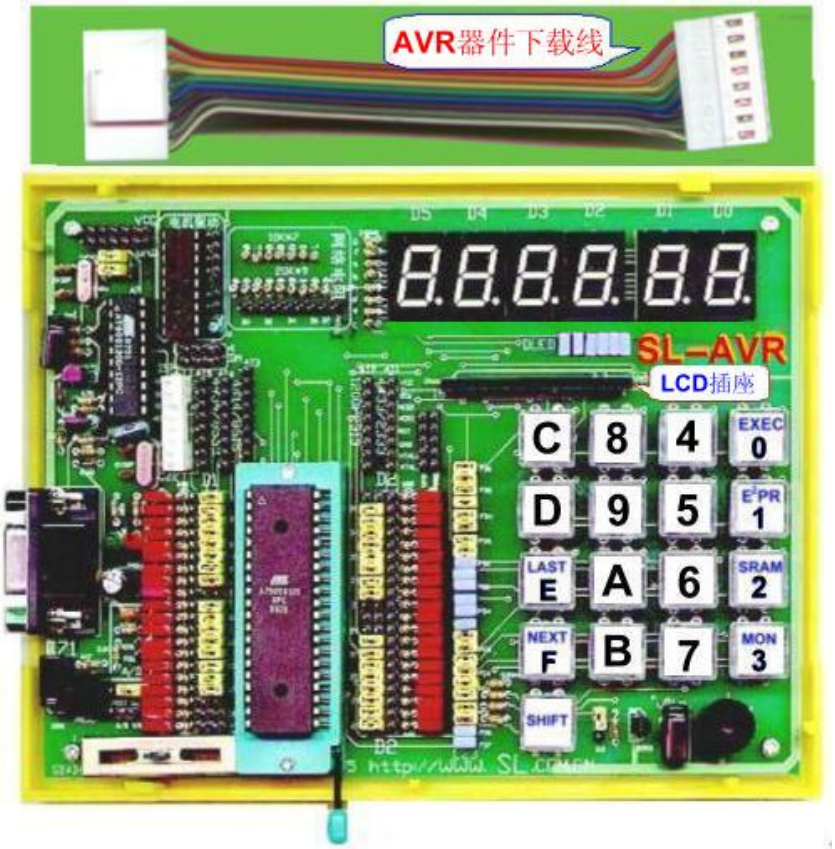


图 3.22　SL-AVR 编程开发实验器实物图

有步进电机驱动电路模块;模拟比较输入电路;音响电路;复位电路;模拟电压输入电路等,随机附 120X170mm 万通实验板及一片 AT90S8515 器件。SL-AVR 适用于所有具有串行下载编程功能的 AVR 单片机,用户板上的 AVR 器件无需拆下即可编程,同时还可做 AVR 单片机的 I/O 口、A/D、D/A、LED、LCD、键盘输入、步进电机控制、音频输出、模拟比较等开发实验;提供功能强大的 WIN 版汇编级编译器 WAVRASM、模拟仿真调试软件 AVRStudio3.X 及串行下载软件 AVR PROG,同时也提供限时版的 C(IAR、Icc)编译器,不限时的(限 2KB)BASCOM-AVR 编译器;对初学 AVR 单片机的设计者,可暂时节省购买较昂贵的实时仿真器及万用编程器的费用。SL-AVR 开发实验器提供的几十个实用实验程序,你也可改变硬件接口,修改程序,实现原程序的功能;这对大专院校学生发挥其

创造性思维及动手能力的培养特别有用,可改变我国传统教育下的“高分低能”的弊病。本开发实验器也可当科研样机使用。

CZ3 引脚功能表

| 引脚 8515 | CZ3 插座 接脚 | CZ3 插座 编号 | CZ3 插座 接脚 | 外引功能 |
|---|---|---|---|---|
| PA0 | ● | 1 | ● | A |
| PA1 | ● | 2 | ● | B |
| PA2 | ● | 3 | ● | C |
| PA3 | ● | 4 | ● | D |
| PA4 | ● | 5 | ● | E |
| PA5 | ● | 6 | ● | F |
| PA6 | ● | 7 | ● | G |
| PA7 | ● | 8 | ● | H |
| PD0 | ● | 9 | ● | D0 |
| PD1 | ● | 10 | ● | D1 |
| PD2 | ● | 11 | ● | D2 |
| PD3 | ● | 12 | ● | D3 |
| PD4 | ● | 13 | ● | D4 |
| PD5 | ● | 14 | ● | D5 |
| PC7 | ● | 15 | ● | J7 |
| PC6 | ● | 16 | ● | J6 |
| PC5 | ● | 17 | ● | J5 |
| PC4 | ● | 18 | ● | J4 |
| PC3 | ● | 19 | ● | J3 |
| PC2 | ● | 20 | ● | J2 |
| PC1 | ● | 21 | ● | J1 |
| PC0 | ● | 22 | ● | J0 |
| PD6 | ● | 23 | ● | NC |
| PD7 | ● | 24 | ● | shift |

### 3.3.2 SL-AVR 硬件接口电路说明如下:

1. CZ1: 电源及通讯下载插座,电源线为地及+5V,通讯电缆一头接 CZ1,另一头接计算机 RS232 九针插座;
2. CZ2: 该列八针的 (ISP)插座,即 AVR 单片机的下载信号插座。

★ 本开发实验器配一片 AT90S8515 器件,绝大多数实验使用该器件,硬件(用短路块)连接也按该器件连接,其它器件作为选购件;

(1) CZ2 (ISP)下载插座图 3.23:引脚功能从上到下分别为 VCC, SCK, MISO, MOSI, RESET, GND, XTAL2, XTAL1;随机附有一条 8 线信号线,由用户接插到对应 AVR 单片机(AT1-AT5)的信号脚上,CZ2:也可接到用户板作 AVR 单片机的串行下载编程用,如用户板有晶振,则 XTAL1/XTAL2 两信号线无需接出;;

(2) AT1 插针座为:AT90S4433/AT90S2333 ISP 下载信号线座;

AT2 插针座为:AT90S1200/AT90S2313 ISP 下载信号线座;

AT3 插针座为:AT90S4414/AT90S8515 ISP 下载信号线座;

AT4 插针座为:AT90S4443/AT90S8535 ISP 下载信号线座;

AT5(90SX1) 插针座,留给用户连线, 接到其它 AVR 单片机下载信号引脚上,作为 ISP 下载信号线座;

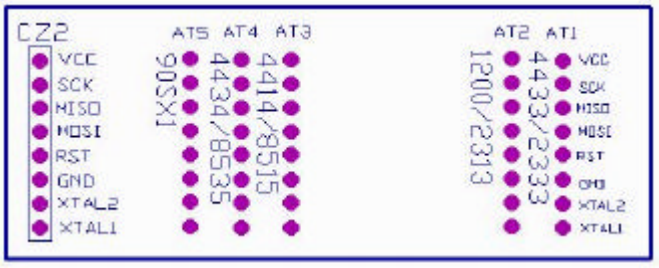


图 3.23 CZ2 下载信号线插座

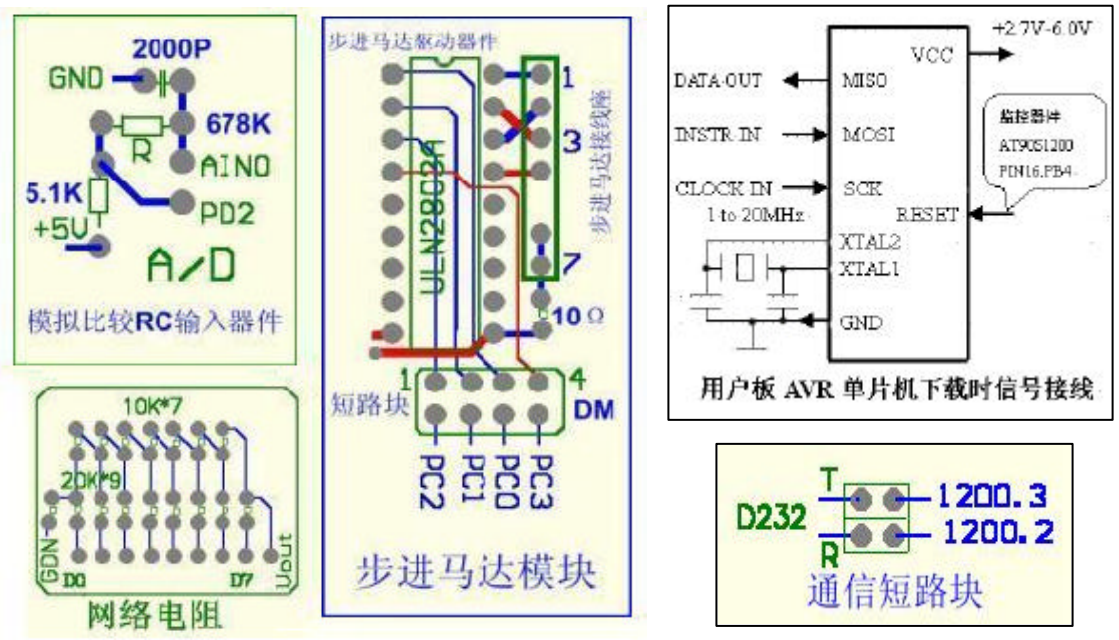


图 3.24

3. D232 通信线短路块图 3.24:插上短路块,接到 AT90S1200,拔出短路块,可从 T、R 端用插针线接到单片机通信口,可做单片机异步通信 UART 或主从同步通信 SPI 或 ISP 下载通信;
4. 电机驱动模块图 3.24:为步进电机驱动模块,ULN2803A 是步进电机驱动集成电路,输出端 1-7 接线柱,可接 4.5V 步进电机(选购件),用户做实验时插上 4 只短路块,本书提供的实验程序用 AT90S8515 的 PC0-PC3 口;
5. 网络电阻模块图 3.24:为了提高模拟比较器的精度,可采用网络电阻,0-5V 精度为±0.02V;
6. A/D 图 3.24:为片内摸拟比较器作 A/D 转换外部元件电路(最好 R 精度为 1%,C 精度为 5%,所接阻值仅供参考),AIN0,PD2 为接线端;
7. LED 模块图 3.25:由六只 LED 数码管组成,D5-D2 可作地址显示,D1-D0 作数据显示,LED 数码管字位口对应接 AT90S8515 的 PD5-PD0 口,LED 数码管的字形(abcdefgh)对应接 AT90S8515 的 PB7-PB0 口,见 CZ3 接线表;

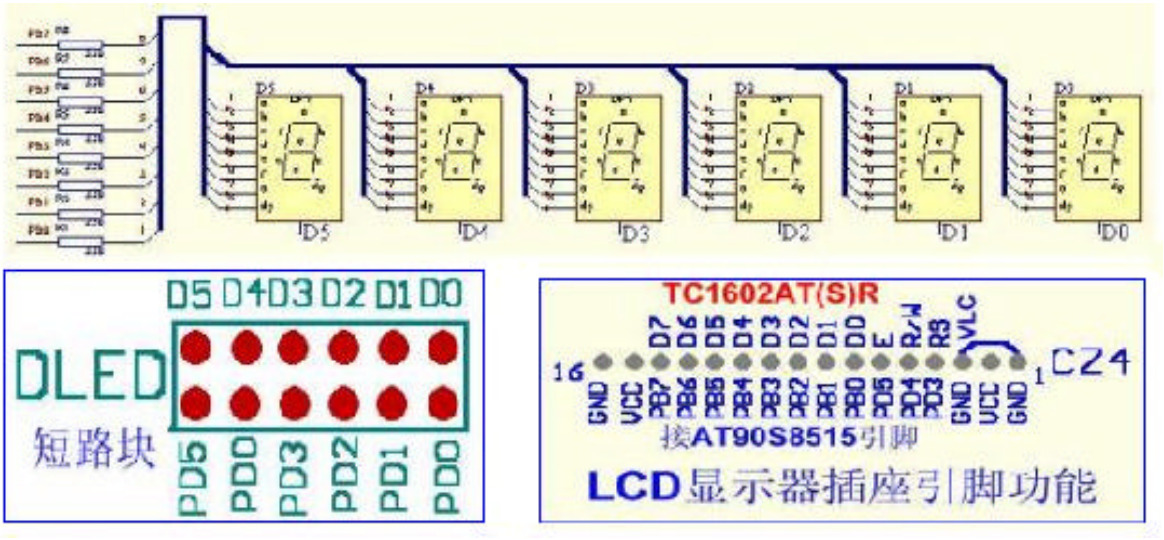


图 3.25

8. CZ3 接线排针:为 AT90S8515 对应引脚 PB0-PB7,PD0-PD7,PC7-PC0,插上短路块接到键盘(17 键)和 LED/LCD 显示器,拔出短路块,用接插线可把键盘与显示改为其它 I/O 口,也可改用其它器件(如 AT90S2313/AT90S8535 等)接到器件相应引脚上(D1、D2 短路块处)。**本开发机提供的几十个实用实验程序,你也可改变硬件接口,修改程序,实现原程序的功能;这对大专院校学生发挥其创造性思维及动手能力的培养特别有用,可改变我国传统教育下的“高分低能”的弊病**;
9. CZ4 接线座图 3.25:为 2 行 X16 字 LCD 液晶显示模块插座,用 AT90S8515 的 PB、PD 口;
10. 17 键键盘模块图 3.26:接 AT90S8515 的 PC 口,16 个数字键,另一个 SHIFT 换档键,当按下 SHIFT 键,同时按数字键,即实现该键上档命令,当然根据程序的要求,这些数字的名称可以重新定义,不仿一试;如果 PC7 端用接插线接地,又 DC0,DC1,DC2,DC3 接到 I/O 口,则 F 键,B 键,7 键,3 键可当按钮开关输入用。

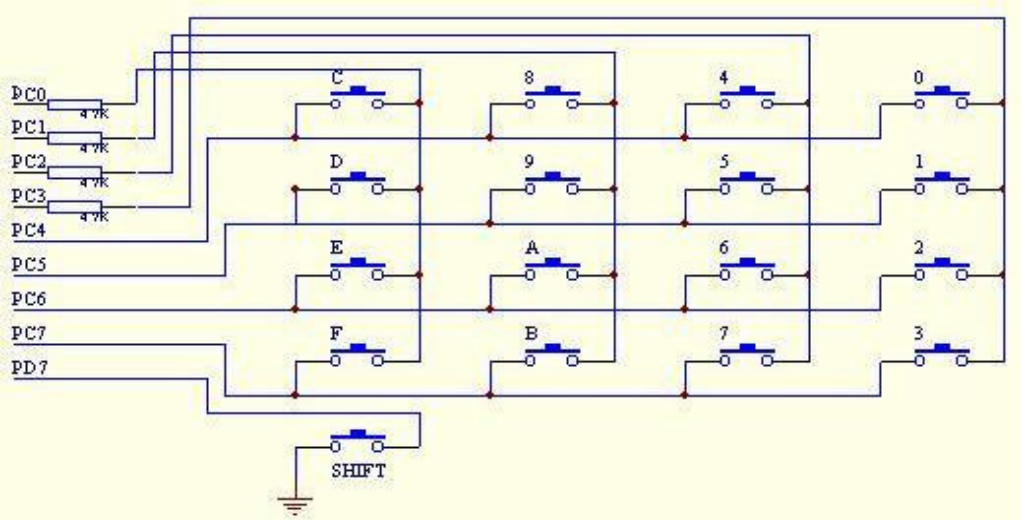


图 3.26

**11. D1、D2 短路块列图** 3.27 **：使单片机引脚作输出或输入用,插上短路块引脚作输出,用** LED **显示高低电平,低电平** LED **灯亮;不插短路块,器件引脚可作输入用或作其它用途,如电源引线等,**

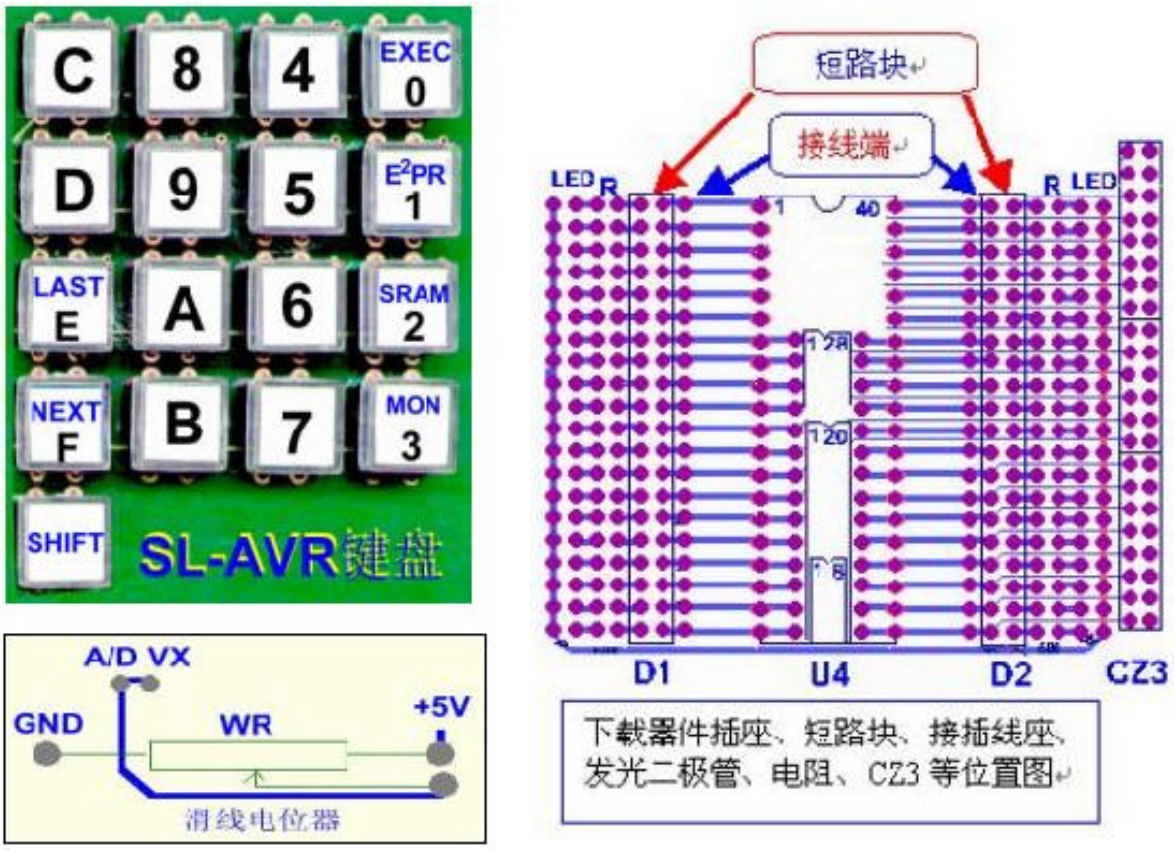


图 3.27

**12. U4 图** 3.27**: 作为** AVR **单片机四种** DIP(DIP8/20/28/40)**封装器件下载插座,器件插放时缺口向上,向下插座底部对齐**;

**13. WR 图** 3.27**:** WR **划线电位器作为模拟信号输入用,两边分别接上** VCC、GND,**中间头**(VX)**用连接线接到单片机作模拟信号输入脚(如** AIN1);

**14. DY 图 3.28: 音响输入端,接相应器件的右下脚(5/11/15/21),插上短路块,接通 AT90S8515 的 PC0 的音响信号,也可用接插线接到单片机任何 I/O 脚上作音响输出;**

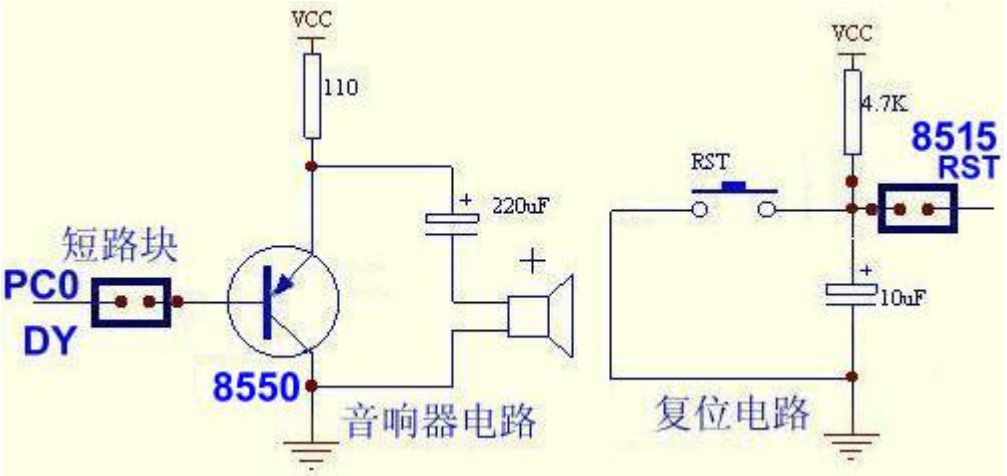


图 3.28

**15. AVR RST 复位按钮图 3.28: 作为程序下载后再启动复位用,一般情况程序下载结束或开机通电时程序就自动执行,可用接插线连接 RST OUT 与对应器件复位脚;本机出厂时/RST OUT 插短块接通,接 AT90S8515 的复位脚;**

**16. POWER: 红色 LED,电源指示; BUSY: 录色 LED,下载通讯工作忙指示;**

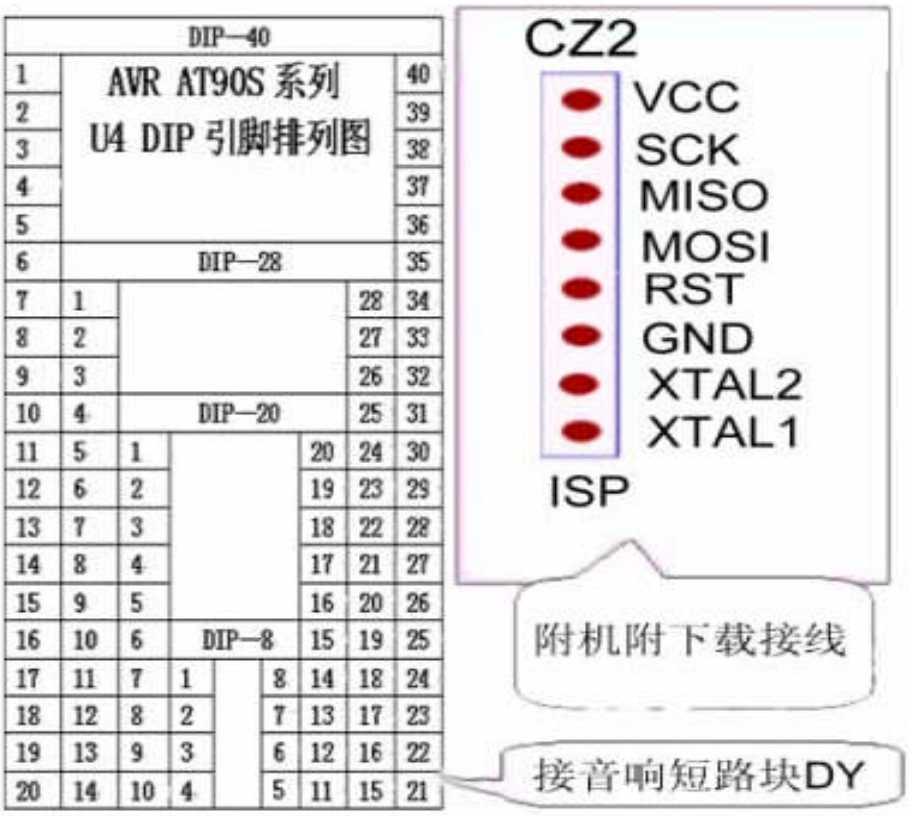


图 3.29 U4 对应 AVR 器件引脚及下载信号线名称

## 3.4 AVR 单片机串行下载操作:

在光盘*:\AVR\AVR\AVR PROG 下，双击图标进入串行下载窗口，也可把图标移到桌面成快捷菜单，双击图标进入串行下载操作窗口。


注意:必须在通电并连接 SL-AVR 万用串行开发实验器情况下才会出现下载窗口!


图 3.30 AVR 下载窗口

## 3.5 SLAVR*.ASM 综合程序简介

程序说明:源程序文件(*.ASM)编译后可供调试(*.OBJ),串行下载(*.HEX);
光盘中文件夹 SLAVR 为源程序文件名,编号对应为第几章几节几例!
综合程序 SLAVR73A.ASM 功能如下:
(1). LED 及 LCD 显示程序,有自动识别 LED 或 LCD 功能
(2). 键盘扫描输入程序,0-F 为 16 个数字键; 还有上档命令键,EXEC--执行键;
E2PROM--读键; SRAM--读写键; MON--返回初始状态键;
LAST--上一单元地址键; NEXT--下一单元地址键;
SHIFT--转换上档命令键,先按 SHIFT 键,再按命令键,就执行上档键的命令;
/RST--复位键,执行程序后,要机器回到初始化状态,必须按复位键;
(3) 按数字键显示对应数字,并有小数点作为光标,提示下一步工作位置,
按命令键(先按 SHIFT)执行相应命令;
(4) 对应功能入口地址(地址数字后零可省)
0070H-01FFH 读、写内部 SRAM(监控规定 SRAM 读写范围)
0000H-01FFH 读片内 E2PROM 数据
0200H-歌曲-祝你生日快乐,万水千山总是情
0300H-LED 上 8 字循环显示
0320H-LED 上 0-F 字符循环显示
0400H-逐次逼近法 A/D 转换(需接网络电阻,另见说明)
0500H-LCD 初始化程序
0700H-LCD 上字符(→ ←)左右移位程序
0740H-LCD 上 0-F 字符循环显示
0800H-LCD 显示 LCD 所有字符
* 程序清单见光盘文件 SLAVR73A.ASM,更多实用实验程序见<<第七章 单片机的应用>>