第四章 AVR 单片机指令系统

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原文翻译,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 下载开发下载实验器上验证通过,把测试实验程序刻在光盘上,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摸得着,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 单片机每条指令对应的实验源程序见文件夹<<指令 ASM>>

计算机的指令系统是一套控制计算机操作的代码,称之为机器语言。计算机只能识别和执行机器语言的指令。为了便于人们理解、记忆和使用,通常用汇编语言指令来描述计算机的指令系统。汇编语言指令可通过汇编器翻译成计算机能识别的机器语言。

AVR 单片机指令系统是 RISC 结构的精简指令集,是一种简明易掌握、效率高的指令系统。

AVR 单片机指令系统速查表,不同器件使用不同的指令表,见附录 3:

(1) 89 条指令器件:AT90S1200,最基本指令;
(2) 90 条指令器件(□):Attiny11/12/15/22; 90 条指令=□+89 条基本指令
(3) 118 条指令器件(◇):AT90S2313/2323/2343/2333,/4414/4433/4434/8515/90S8534/8535 ;118 条指令=◇+ 90 条;
(4) 121 条指令器件(△):ATmega603/103; 121 条指令=△+ 118 条;
(5) 130 条指令器件(☆):ATmega161; 130 条指令=☆+121 条

AVR 大多数执行时间为单个时钟周期。这一章主要分析 AVR 单片机指令系统的功能和使用方法。下表为常用 AVR 器件指令表:

AVR 器件(指令速查表) 118 条指令器件

AT90S2313/2323/2343/2333 ,AT90S4414/4433/4434/8515,AT90S8534/8535

算术和逻辑指令		BRCC k	C 清零转	位指令和位测试指令	
ADD Rd,Rr	加法	BRSH k	≥转	SBI P,b	置位 I/O 位
ADC Rd,Rr	带进位加	BRLO k	小于转(无符号)	CBI P,b	清零 I/O 位
◇ ADIW Rdl,K	加立即数	BRMI k	负数转移	LSL Rd	左移
SUB Rd,Rr	减法	BRPL k	正数转移	LSR Rd	右移
SUBI Rd,Rr	减立即数	BRGE k	≥转(带符号)	ROL Rd	带进位左循环
SBC Rd,Rr	带进位减	BRLT k	小于转(带符号)	ROR Rd	带进位右循环
SBCI Rd,K	带 C 减立即数	BRHS k	H 置位转移	ASR Rd	算术右移
◇ SBIW Rdl,K	减立即数	BRHC k	H 清零转移	SWAP Rd	半字节交换
AND Rd,Rr	与	BRTS k	T 置位转移	BSET s	置位 SREG
ANDI Rd,K	与立即数	BRTC k	T 清零转移	BCLR s	清零 SREG
OR Rd,Rr	或	BRVS k	V 置位转移	BST Rr,b	Rr 的 b 位送 T
ORI Rd,K	或立即数	BRVC k	V 清零转移	BLD Rd	T 送 Rr 的 b 位
EOR Rd,Rr	异或	BRIE k	中断位置位转移	SEC	置位 C
COM Rd	取反	BRID k	中断位清零转移	CLC	清零 C
NEG Rd	取补	数据传送指令		SEN	置位 N
SBR Rd,K	寄存器位置位	MOV Rd,Rr	寄存器传送	CLN	清零 N
CBR Rd,K	寄存器位清零	◇ LDI Rd,Rr	装入立即数	SEZ	置位 Z
INC Rd	加 1	◇ LD Rd, X	X 间接取数	CLZ	清零 Z
DEC Rd	减 1	◇ LD Rd, X+	X 间接取数后+	SEI	置位 I
TST Rd	测试零或负	◇ LD Rd,-X	X 间接取数先–	CLI	清零 I
CLR Rd	寄存器清零	◇ LD Rd,Y	Y 间接取数	SES	置位 S
SER Rd	寄存器置 FF	◇ LD Rd,Y+	Y 间接取数后+	CLS	清零 S
条件转移指令		◇ LD Rd,–Y	Y 间接取数先–	SEV	置位 V
RJMP k	相对转移	◇ LDD Rd,Y+q	Y 间接取数+ q	CLV	清零 V
◇ IJMP	间接转移(Z)	LD Rd, Z	Z 间接取数	SET	置位 T
RCALL k	相对调用	◇ LD Rd, Z+	Z 间接取数后+	CLT	清零 T
◇ ICALL	间接调用(Z)	◇ LD Rd,–Z	Z 间接取数先–	SEH	置位 H
RET	子程序返回	◇ LDD Rd, Z+q	Z 间接取数+ q	CLH	清零 H
RETI	中断返回	◇ LDS Rd,K	从 SRAM 装入	NOP	空操作
CPSE Rd,Rr	比较相等跳行	ST X ,Rr	X 间接存数	SLEEP	休眠指令
CP Rd,Rr	比较	◇ ST X+ ,Rr	X 间接存数后+	WDR	看门狗复位
CPC Rd,Rr	带进位比较	◇ ST –X ,Rr	X 间接存数先–	90 条指令为 Attiny11/12/15/22= □+89 条基本指令是 AT90S1200	
CPI Rd,K	与立即数比较	◇ ST Y ,Rr	Y 间接存数		
SBRC Rr,b	位清零跳行	◇ ST Y+ ,Rr	Y 间接存数后+		
SBRS Rr,b	位置位跳行	◇ ST –Y ,Rr	Y 间接存数先–		
SBIC P,b	I/O 位清零跳行	◇ STD Y+q ,Rr	Y 间接存数+ q		
SBIS P,b	I/O 位置位跳行	ST Z ,Rr	Z 间接存数		
BRBS s,k	SREG 位置位转	◇ ST Z+ ,Rr	Z 间接存数后+	118 条指令器件= ◇+ 90 条指令器件	
BRBC s,k	SREG 位清零转	◇ ST –Z ,Rr	Z 间接存数先-		
BREQ k	相等转移	◇ STD Z+q ,Rr	Z 间接存数+q		
BRNE k	不相等转移	◇ STS k,,Rr	数据送 SRAM		
BRCS k	C 置位转	□ LPM	从程序区取数		
		IN Rd,P	从/O 口取数		
		OUT P, Rdr	存数 I/O 口		
		PUSH Rr	压栈		
		POP Rd,	出栈		

4.1 指令格式

4.1.1 汇编指令

汇编语言源文件是由汇编语言代码和汇编程序指令所组成的 ASCII 字符文件。

一、汇编语言源文件

汇编语言源文件包括指令助记符、标号和伪指令。指令助记符和伪指令常带操作数。

每条程序输入行首先是标号,标号为字母数字串，并带一个冒号。使用标号的目的是为了跳转和转移指令及在程序存储器和 SRAM 中定义变量名。

程序输入行有下列四种形式:

（1)【标号:】伪指令【操作数】【注释义】

（2）【标号:】指令【操作数】【注释】

(3)注释

（4）空行

注释有下列形式:【文字】

括号内的项是任选的。用于注释的分号（;）及到行结尾的文字，汇编器是忽略的。标号、指令和伪指令在后面有详细说明。

例子:

```
Label：.EQU Var1=100    ；置 Var1 等于 100（伪指令）
       .EQU Var2=200    ；置 Var2 等于 200
test:  rjmp  test       ；无限循环（指令）
                        ：纯注释行
                        ；另一个注释行
```

注意: 不限制有关标号、伪指令、注释或指令的列位置。

二、指令助记符

汇编器认可指令集中的指令助记符。指令集中综合了助记符并给出了参数。

操作数有下列形式:

Rd：R0～R31 或 R16～R31（取决于指令)。

Rr：R0～R31。

b：常数（0～7)，可能是常数表达式。

S：常数（0～7)，可能是常数表达式。

p：常数(0～31／63）可能是常数表达式。

K：常数（0～255）可能是常数表达式。

k：常数，值范围取决于指令，可能是常数表达式。

q：常数（0～63)，可能是常数表达式。

4.1.2 汇编器伪指令

汇编器提供一些伪指令，伪指令并不直接转换成操作数，而是用于调整存储器中程序的位置、定义宏、初始化存储器等。全部伪指令在表 4.2 中给出。

1. BYTE——保存字节到变量

BYTE 伪指令保存存储的内容到 SRAM 中。为了能提供所要保存的位置，BYTE 伪指令前应有标号。该伪指令带一个表征被保存字节数的参数。该伪指令仅用在数据段内（见伪指令 CSEG，DSEG 和 ESEG)。注意：必须带一个参数，字节数的位置不需要初始化。

语法: LABEL:. BYTE 表达式

2. CSEG——代码段

CSEG 伪指令定义代码段的开始位置。一个汇编文件包含几个代码段，这些代码段在汇 编时被连接成一个代码段。在代码段中不能使用 BYTE 伪指令，典型的缺省段为代码段。代码段有一个字定位计数器。ORG 伪指令用于放置代码段和放置程序存储器指定位置的常数。

CSEG 伪指令不带参数。

语法：. CSEG

3. DB——在程序存储器或 EEPROM 存储器中定义字节常数

DB 伪指令保存数据到程序存储器或 EEPROM 存储器中。为了提供被保存的位置， 在 DB 伪指令前必须有标号。DB 伪指令可带一个表达式表， 至少有一个表达式。DB 伪指令必须放在代码段或 EEPROM 段。表达式表是一系列表达式，用逗号分隔。每个表达式必须是一 128～255 之间的有效值。如果表达式有效值是负数， 则用 8 位 2 的补码放在程序存储器或 EEPROM 存储器中。如果 DB 伪指令用在代码段， 并且表达式表多于一个表达式， 则以两个字节组合成一个字放在程序存储器中。如果表达式表是奇数， 那么最后一个表达式将独自以字格式放在程序存储器中， 而不管下一行汇编代码是否是单个 DB 伪指令。

语法：LABEL:. DB 表达式

4. DEF——设置寄存器的符号名

DEF 伪指令允许寄存器用符号代替。一个定义的符号用在程序中，并指定一个寄存器,一个寄存器可以赋几个符号。符号在后面程序中能再定义。

语法：. DEF 符号=寄存器

5. DEVICE——定义被汇编的器件

DEVICE 伪指令允许用户告知汇编器被执行的代码使用那种器件。如果使用该伪指令，若在代码中有指定的器件不提供的指令，则提示一个警告。如果代码段或 EEPROM 段的尺寸大于被指定器件的尺寸， 也提示警告。如果不使用 DEVICE 伪指令，则假定器件提供所有的指令， 也不限制存储器尺寸。

语法： .DEVICE AT90S1200 AT90S2313 AT90S4414 AT90S8515

6. DSEG —数据段

DSEG 伪指令定义数据段的开始。一个汇编文件能包含几个数据段，这些数据段在汇编时被连接成一个数据段。一个数据段正常仅由 BYTE 伪指令（和标号）组成。数据段有自己的定位字节计数器。ORG 伪指令被用于在 SRAM 指定位置放置变量。DSEG 伪指令不带参数。

语法： .DSEG

7. DW—在程序存储器和 EEPROM 存储器中定义字常数

DW 伪指令保存代码到程序存储器或 EEPROM 存储器，为了提供被保存的位置，在 DW 伪指令前必须有标号。DW 伪指令可带一个表达式表， 至少有一个表达式。DW 伪指令必须放在代码段或 EEPROM 段。表达式表是一系列表达式， 用逗号分隔。每个表达式必须是一 32 768～65 535 之间的有效值。如果表达式有效值是负数， 则用 16 位 2 的补码放在程序存储器中。

语法：LABEL： .DW 表达式表

8·ENDMACRO —宏结束

ENDMACRO 伪指令定义宏定义的结束。该伪指令并不带参数，参见 MACRO 宏定义伪指令。

语法： .ENDMACRO

9. EQU—设置一个符号相等于一个表达式

EQU 伪指令赋一个值到标号，该标号用于后面的表达式， 用 EQU 伪指令赋值的标号是一个常数， 不能改变或重定义。

语法： .EQU 标号= 表达式

10. ESEG - EEPROM 段

ESEG 伪指令定义 EEPROM 段的开始位置。一个汇编文件包含几个 EEPROM 段，这些 EEPROM 段在汇编时被连接成一个 EEPROM 段。在 EEPROM 段中不能使用 BYTE 伪指令。EEPROM 段有一个字节定位计数器。ORG 伪指令用于放置 EEPROM 存储器指定位置的常数。ESEG 伪指令不带参数。

语法： .ESEG

11. EXIT—退出文件

EXIT 伪指令告诉汇编器停止汇编该文件。正常情况下，汇编器汇编到文件的结束。如果 EXIT 出现在包括文件中，则汇编器从文件中 INCLUDE 伪指令行继续汇编。

语法： .EXIT

12·INCLUDE—包括另外的文件

INCLUDE 伪指令告诉汇编器从指定的文件开始读。然后汇编器汇编指定的文件，直到文件结束或遇到 EXIT 伪指令。一个包括文件也可能自己用 INCLUDE 伪指令来表示。

语法： .INCLUDE“文件名”

13. LIST—打开列表文件生成器

LIST 伪指令告诉汇编器打开列表文件生成器。汇编器生成一个汇编源代码、地址和操作代码的文件列表。列表文件生成器缺省值是打开。该伪指令总是与 NOLIST 伪指令一起出现，用于生成列表或汇编源文件有选择的列表。

语法： .LIST

14 LISTMAC—打开宏表达式

LISTMAC 伪指令告诉汇编器，当调用宏时，用列表生成器在列表文件中显示宏表达式。缺省值仅是在列表文件中显示宏调用参数。

语法： .LISTMAC

15. MACRCO 一宏开始

MACRO 伪指令告诉汇编器这是宏开始。MACRO 伪指令带宏名和参数。当后面的程序中写了宏名，被表达的宏程序在指定位置被调用。一个宏可带 10 个参数。这些参数在宏定义中用@0～@9 代表。当调用一个宏时，参数用逗号分隔。宏定义用 ENDMACRO 伪指令结束。缺省值为汇编器的列表生成器，仅列表宏调用。为了在列表文件中包括宏表达式，必须使用 LISTMAC 伪指令。在列表文件的操作代码域内宏用 a+ 作记号。

语法： .MACRO 宏名

16. NOLIST--关闭列表文件生成器

NOLIST 伪指令告诉汇编器关闭列表文件生成器。正常情况下，汇编器生成一个汇编源代码、地址和操作代码文件列表。缺省时为打开列表文件，但是可用该伪指令禁止列表。为了使被选择的汇编源文件部分产生列表文件，该伪指令可以与 LIST 伪指令一起使用。

语法： .NOLIST

17. ORG 一设置程序起始位置

ORG 伪指令设置定位计数器一个绝对值。设置的值为一个参数。如果 ORG 伪指令放在数据段，则设置 SRAM 定位计数器；如果该伪指令放在代码段，则设置程序存储器计数器；如果该伪指令放在 EEPROM 段，则设置 EEPROM 定位计数器。如果该伪指令前带标号（在相同的源代码行），则标号由参数值给出。代码和 EEPROM 定位计数器的缺省值是零；而当汇编启动时，SRAM 定位计数器的缺省值是 32（因为寄存器占有地址为 0～ 31）。注意，EEPROM 和 SRAM 定位计数器按字节计数，而程序存储器定位计数器按字计数。

语法： .ORG 表达式

18. SET一设置一个与表达式相等的符号

SET 伪指令赋值给一个标号。这个标号能用在后面的表达式中。用 SET 伪指令赋值的标号在后面的程序中能改变。

语法： .SET 标号 = 表达式

4·1．3 表达式

汇编器包括一些表达式，表达式由操作数、运算符和函数组成。所有的表达式内部为 32 位。

一、操作数

（1）用户定义的标号，该标号给出了放置标号位置的定位计数器的值。
（2）用户用 SET 伪指令定义的变量。
（3）用户用 EQU 伪指令定义的常数。
（4）整数常数，包括下列几种形式：
·十进制（缺省值）： 10， 255
·十六进制数（二进制表示法）：0x0a,$0a,0xff.$ff
二进制数：0b00001010,0bllllllll
（5）PC：程序存储器定位计数器的当前值。

二、 函 数

（1）LOW（表达式）返回一个表达式的低字节。
（2）HIGH（表达式）返回一个表达式的第二个字节。
（3）BYTE2（表达式）与 HIGH 函数相同。
（4）BYTE3（表达式）返回一个表达式的第三个字节。
（5）BYTE4（表达式）返回一个表达式的第四个字节。
（6）LWRD（表达式）返回一个表达式的 0～ 15 位。
(7) HWRD（表达式）返回一个表达式的 16～ 31 位。
（8）PAGE（表达式）返回一个表达式的 16～ 21 位。
（9）EXP2（表达式）返回 2^表达式。
（10）LOG2（表达式）返回 LOG2（表达式）的整数部分。

三、 运算符

汇编器提供的部分运算符见表 3.3。越高的运算符，优先级越高。表达式可以用括号括起来，并且与括号外任意表达式所组合的表达式总是有效的。

表4.3 部分运算符表

序号	名称	符号	优先级	说明
1	逻辑非	!	14	一元运算符，表达式是 0 返回 1，而表达式为非 0 则返回 0
2	逐位非	~	14	一元运算符，输入表达式的所有位倒置
3	负号	-	14	一元运算符，使表达式为算术负
4	乘法	*	13	二进制运算符，两个表达式相乘
5	除法	/	13	二进制运算符，左边表达式除以右边表达式，得整数的商值
6	加法	+	12	二进制运算符，两个表达式相加
7	减法	-	12	二进制运算符，左边表达式减去右边表达式
8	左移	<<	11	二进制运算符，左边表达式左移右边表达式给出的次数
9	右移	>>	11	二进制运算符，左边表达式右移右边表达式给出的次数
10	小于	<	10	二进制运算符，左边带符号表达式小于右边带符号表达式，则为 1，否则为 0
11	小于等于	<=	10	二进制运算符，左边带符号表达式小于或等于右边带符号表达式，则为 1，否则为 0
12	大于	>	10	二进制运算符，左边带符号表达式大于右边带符号表达式，则为 1，否则为 0
13	大于等于	>=	10	二进制运算符，左边带符号表达式大于或等于右边带符号表达式，则为 1，否则为 0
14	等于	==	9	二进制运算符，左边带符号表达式等于右边带符号表达式，则为 1，否则为 0
15	不等于	!=	9	二进制运算符，左边带符号表达式不等于右边带符号表达式，则为 1，否则为 0
16	逐位与	&	8	二进制运算符，两个表达式之间逐位与
17	逐位异或	^	7	二进制运算符，两个表达式之间逐位异或
18	逐位或	|	6	二进制运算符，两个表达式之间逐位或
19	逻辑与	&&	5	二进制运算符，两个表达式逻辑与，非 0 则为 1，否则为 0
20	逻辑或	||	4	二进制运算符，两个表达式逻辑或，非 0 则为 1，否则为 0