第七章　AVR 单片机的应用

ATMEL 公司的 AVR 单片机,是增强型 RISC 内载 Flash 的单片机。芯片上的 Flash 存储器附在用户的产品中,可随时编程、再编程,使用的产品设计容易,更新换代方便。AVR 单片机采用增强的 RISC 结构,使其具有高速处理能力,在一个时钟周期内可执行复杂的指令,每 MHz 可实现 1MIPS 的处理能力。AVR 单片机工作电压为 2.7~6.0V,可以实现耗电最优化。AVR 的单片机广泛应用于计算机外部设备,工业实时控制,仪器仪表,通讯设备,家用电器,宇航设备等各个领域。

本书所提供的实用、实验程序均是在 SL-AVR 开发下载实验器上汇编、调试、下载验证通过的,用户可以放心地学习修改、移植,今后我们还将从网上或电子书光盘形式不断向读者、用户提供实用、实验程序的软、硬件资料。

应用实验源程序见文件夹<<SLAVR>>,应用例子*.ASM,必须编译生成*.OBJ 文件才可调试,如要修改*.ASM,必须修改文件属性,去掉*.ASM 只读文件属性。

7.1.0　通用延时子程序

```
;**********************AVR 单片机实用程序 ************************
;* 标题:通用延时子程序,文件名:DELAY.ASM
;* 版本:     1.0
;* 最后更新日期:2000.09.10
;* 支援 E-mail: gzsl@sl.com.cn
;* 描述
;*  利用寄存器内容减 1 不 0 为转的多级嵌套,只需改变一个寄存器延时常数,
;* 就可改变延时时间
;* 作者: SL. Z
;* 程序适用于有 SRAM 的 AVR 单片机
;*********************************************************************
.include "8515def.inc"       ;器件配置文件,决不可少,不然汇编通不过
.DEF   TEMP1    =R20
.DEF   CON      =R21
.org  $0000
    rjmp    RESET            ;复位
.ORG$0010                    ;跳过中断区
RESET:  ldi r16,high(RAMEND)     ;设 AT90S8515 堆栈 为$025F
    OUT SPH,r16                  ;见器件配置文件"8515def.inc "
    ldi r16,low(RAMEND)
    OUT     SPL,R16
    ser temp1
    SER     CON                  ;temp1 直接置数$FF, A 口
    out DDRA,temp1               ;方向寄存器设定 A 口为输出
    LDI     R16,0X70
LOOP:  OUT     PORTA,TEMP1
        RCALL   DELAY            ;调用通用延时子程序
        EOR     TEMP1,CON        ; 异或
```

```
        RJMP    LOOP
;注意:以后程序中的通用延时子程序从略!
delay:                      ; 通用延时子程序
        push    r16         ;进栈需 2t
L0:     push    r16         ;进栈需 2t
L1:     push    r16         ;进栈需 2t
L2:     push    r16         ;进栈需 2t
L3:     dec     r16         ; -1 需 1t
        brne    L3          ;不为 0 转,为 0 顺执, 需 1t/2t
        pop     r16         ;出栈 需 2t
        dec     r16         ;-1 需 1t
        brne    L2          ; 不为 0 转,为 0 顺执,需 1t/2t
        pop     r16         ; 出栈需 2t
        dec     r16         ; -1 需 1t
        brne    L1          ; 不为 0 转,为 0 顺执,需 1t/2t
        pop     r16         ; 出栈需 2t
        dec     r16         ;-1 需 1t
        brne    L0          ; 不为 0 转,为 0 顺执,需 1T/2T
        pop     r16         ; 出栈需 2t
        ret                 ; 子程序返回需 4t
```

一次嵌套循环公式 T 机器周期数,t 延时时间,AVR 8MHz 晶振时,每个机器周期为 0.125μS

$$T=7X-1+\sum_{1}^{x}(3x-1) \qquad ;t=T*(1\text{ 个机器周期时间})$$

二次嵌套循环公式

$$T=7X-1+\sum_{i=1}^{x}(7x-1)+\sum_{i=1}^{x}\sum_{i=1}^{x}(3x-1)$$

三次嵌套循环公式

$$T=7X-1+\sum_{i=1}^{x}(7x-1)+\sum_{i=1}^{x}\sum_{i=1}^{x}(7x-1)+\sum_{i=1}^{x}\sum_{i=1}^{x}\sum_{i=1}^{x}(3x-1)$$

```
;*****************************************************************
;*   二次嵌套通用延时程序                                        *
;*                                                               *
;* if fos=8mhz         time (3.5us------1s)                      *
;*       dt                time                                *
;*       22                 1ms                                *
;*       29                 2ms                                *
```

```
;*        40                  5ms                                          *
;*        51                  10ms                                         *
;*        65                  20ms                                         *
;*        90                  50ms                                         *
;*        114                 100ms                                        *
;*        144                 200ms                                        *
;*        197                 500ms                                        *
;*        249                  1s                                          *
;*********************************************************************
delay:   push    dt
del1:    push    dt
del2:    push    dt
del3:    dec     dt
         brne    del3
         pop     dt
         dec     dt
         brne    del2
         pop     dt
         dec     dt
         brne    del1
         pop     dt
         ret
;*********************************************************************
;*    一次循环通用延时程序                                             *
;*                                                                     *
;* if  fos=8mhz          time  (3.5us------10ms)                       *
;*        dt                 time                                      *
;*        71                  1ms                                      *
;*        101                 2ms                                      *
;*        161                 5ms                                      *
;*        228                 10ms                                     *
;*********************************************************************
delay:   push    dt
del2:    push    dt
del3:    dec     dt
           brne    del3
           pop     dt
           dec     dt
           brne    del2
           pop     dt
           ret
```

三次嵌套通用延时程序,在 8MHz 晶振下测试数据,H 为十六进制,D 为十进制

通用延时子程序时间常数所对应的延时周期数及时间见下表

H	D	T 周期数	t	H	D	T 周期数	t
1	1	28	3.50 μs	21	33	219556	27.4445 ms
2	2	76	9.50 μs	22	34	244756	30.5945
3	3	166	20.75	23	35	272062	34.00775
4	4	316	39.50	24	36	301588	37.6985
5	5	547	68.38	25	37	333451	41.68138
6	6	883	110.38	26	38	367771	45.97138
7	7	1351	168.88	27	39	404671	50.58388 ms
8	8	1981	247.63	28	40	444277	55.53463
9	9	2806	350.75	29	41	486718	60.83975
A	10	3862	482.75	2A	42	532126	66.51575
B	11	5200	0.65 ms	2B	43	580636	72.5795
C	12	6800	0.85 ms	2C	44	632386	79.04825
D	13	8800	1.10 ms	2D	45	687517	85.93963
E	14	11221	1.4	2E	46	746173	93.27163
F	15	14077	1.76	2F	47	808501	101.0626 ms
10	16	17443	2.18 ms	30	48	874651	109.3314
11	17	21376	2.67	31	49	944776	118.097
12	18	25936	3.24 ms	32	50	1019032	127.379
13	19	31186	3.90	33	51	1097578	137.1973
14	20	37192	4.65	34	52	1180576	147.572
15	21	44023	5.50 ms	35	53	1268191	158.5239
16	22	51751	6.47	36	54	1360591	170.0739
17	23	60451	7.56	37	55	1457947	182.2434
18	24	70201	8.78	38	56	1560433	195.0541 ms
19	25	81082	10.14 ms	39	57	1668226	208.5283
1A	26	93178	11.65	3A	58	1781506	222.6883
1B	27	106576	13.32	3B	59	1900456	237.557
1C	28	121366	15.17	3C	60	2025262	253.1578 ms
1D	29	137641	17.21	3D	61	2156113	269.5141
1E	30	155497	19.44	3E	62	2293201	286.6501
1F	31	175033	21.88	3F	63	2436721	304.5901 ms
20	32	196351	24.54	40	64	2586871	323.36

H	D	T 周期数	t(秒)	H	D	T 周期数	t(秒)
41	65	2743852	0.3429815	61	97	12733880	1.591735
42	66	2907868	0.3634835	62	98	13248680	1.656085
43	67	3079126	0.3848908	63	99	13778930	1.722366
44	68	3257836	0.4072295	64	100	14324930	1.790617
45	69	3444211	0.4305264	65	101	14887000	1.860875
46	70	3638467	0.4548084	66	102	15465450	1.933181
47	71	3840823	0.4801029	67	103	16060590	2.007574 S
48	72	4051501	0.5064376 S	68	104	16672740	2.084093
49	73	4270726	0.5338408	69	105	17302220	2.162778
4A	74	4498726	0.5623408	6A	106	17949360	2.24367
4B	75	4735732	0.5919665	6B	107	18614480	2.326809
4C	76	4981978	0.6227473	6C	108	19297910	2.412238
4D	77	5237701	0.6547126	6D	109	19999980	2.499998
4E	78	5503141	0.6878926	6E	110	20721040	2.59013
4F	79	5778541	0.7223176	6F	111	21461410	2.682677
50	80	6064147	0.7580184	70	112	22221450	2.777682
51	81	6360208	0.795026	71	113	23001500	2.875187
52	82	6666976	0.833372	72	114	23801900	2.975237
53	83	6984706	0.8730882	73	115	24623000	3.077875 S
54	84	7313656	0.914207	74	116	25465170	3.183146
55	85	7654087	0.9567609	75	117	26328750	3.291094
56	86	8006263	1.00078 S	76	118	27214110	3.401764
57	87	8370451	1.04631	77	119	28121610	3.515202
58	88	8746921	1.09337	78	120	29051620	3.631452
59	89	9135946	1.141993	79	121	30004500	3.750562
5A	90	9537802	1.192225	7A	122	30980630	3.872578
5B	91	9952768	1.244096	7B	123	31980380	3.997547 S
5C	92	10381130	1.297641	7C	124	33004130	4.125516
5D	93	10823160	1.352895	7D	125	34052260	4.256532
5E	94	11279160	1.409895	7E	126	35125150	4.390644
5F	95	11749420	1.468677	7F	127	36223200	4.5279
60	96	12234220	1.52928	80	128	37346790	4.668349

H	D	T 周期数	t(秒)	H	D	T 周期数	t(秒)
81	129	38496320	4.812039	A1	161	91483580	11.43545
82	130	39672170	4.959022	A2	162	93729070	11.71613
83	131	40874760	5.109345	A3	163	96015650	12.00196
84	132	42104480	5.26306	A4	164	98343790	12.29297
85	133	43361730	5.420217	A5	165	10071400	12.58925
86	134	44646930	5.580866	A6	166	10312680	12.89085
87	135	45960490	5.745061	A7	167	10558280	13.19784
88	136	47302810	5.912851	A8	168	10808230	13.51028
89	137	48674330	6.084291	A9	169	11062590	13.82824
8A	138	50075450	6.259431	AA	170	11321410	14.15177
8B	139	51506600	6.438324	AB	171	11584760	14.48094
8C	140	52968200	6.621025	AC	172	11852660	14.81583
8D	141	54460690	6.807587	AD	173	12125190	15.15649
8E	142	55984500	6.998063	AE	174	12402390	15.50299
8F	143	57540060	7.192507	AF	175	12684320	15.8554
90	144	59127810	7.390976	B0	176	12971020	16.21378
91	145	60748190	7.593524	B1	177	13262560	16.5782
92	146	62401650	7.800206	B2	178	13558980	16.94873
93	147	64088620	8.011078	B3	179	13860350	17.32543
94	148	65809580	8.226197	B4	180	14166710	17.70838
95	149	67564950	8.445619	B5	181	14478120	18.09765
96	150	69355210	8.669401	B6	182	14794632	18.49329
97	151	71180800	8.8976	B7	183	15116312	18.89539
98	152	73042200	9.130275	B8	184	15443208	19.30401
99	153	74939860	9.367483	B9	185	15775376	19.71922
9A	154	76874260	9.609283	BA	186	16112872	20.14109
9B	155	78845870	9.855734	BB	187	16455760	20.5697
9C	156	80855160	10.1069	BC	188	16804088	21.00511
9D	157	82902600	10.36283	BD	189	17157912	21.44739
9E	158	84988680	10.62358	BE	190	17517296	21.89662
9F	159	87113880	10.88923	BF	191	17882296	22.35287
A0	160	89278690	11.15984	C0	192	18252976	22.81622

H	D	T 周期数	t(秒)	H	D	T 周期数	t(秒)0
C1	193	18629384	23.28673	E1	225	34067096	42.58387
C2	194	19011584	23.76448	E2	226	34667464	43.33433
C3	195	19399632	24.24954	E3	227	35275736	44.09467
C4	196	19793592	24.74199	E4	228	35891968	44.86496
C5	197	20193528	25.24191	E5	229	36516248	45.64531
C6	198	20599488	25.74936	E6	230	37148632	46.43579
C7	199	21011536	26.26442	E7	231	37789200	47.2365
C8	200	21429736	26.78717	E8	232	38438008	48.04751
C9	201	21854144	27.31768	E9	233	39095136	48.86892
CA	202	22284832	27.85604	EA	234	39760656	49.70082
CB	203	22721848	28.40231	EB	235	40434640	50.5433
CC	204	23165264	28.95658	EC	236	41117152	51.39644
CD	205	23615136	29.51892	ED	237	41808272	52.26034
CE	206	24071536	30.08942	EE	238	42508056	53.13507
CF	207	24534512	3066814	EF	239	43216600	54.02075
D0	208	25004136	31.25517	F0	240	43933960	54.91745
D1	209	25480464	31.85058	F1	241	44660216	55.82527
D2	210	25963568	32.45446	F2	242	45395440	56.7443
D3	211	26453512	33.06689	F3	243	46139704	57.67463
D4	212	26950360	33.68795	F4	244	46893072	58.61634
D5	213	27454168	34.31771	F5	245	47655640	59.56955
D6	214	27965008	34.95626	F6	246	48427464	60.53433
D7	215	28482944	35.60368	F7	247	49208624	61.51078
D8	216	29008040	36.26005	F8	248	49999200	62.499
D9	217	29540360	36.92545	F9	249	50799264	63.49908
DA	218	30079976	37.59997	FA	250	51608888	64.51111
DB	219	30626952	38.28369	FB	251	52428152	65.53519
DC	220	31181352	38.97669	FC	252	53257136	66.57142
DD	221	31743248	39.67906	FD	253	54095904	67.61988
DE	222	32312704	40.39088	FE	254	54944544	68.68068
DF	223	32889776	41.11222	FF	255	55803128	69.75391
E0	224	33474552	41.84319	00	256	56671736	70.83967
				$00 为延时最长,因为$00-1=$FF			

7.2　简单 I/O 口输出实验

(1) SLAVR721.ASM ： 测试验证 DIP20 AVR 单片机 B 口、D 口引脚输出和 SL-AVR 开发下载实验器功能,LED 逐位移位,移位速度会变化

```
; AT90S1200 引脚图  "*"表示引脚接 LED 发光二极管
;                  "↓"表示灯亮移位方向
;   /RST   。1    20。VCC
;   PD0 †*。        。* ↓  PB7
;   PD1 †*。        。* ↓  PB6
;   XTAL2  。        。* ↓  PB5
;   XTAL1  。        。* ↓  PB4
;   PD2 †*。        。* ↓  PB3
;   PD3 †*。        。* ↓  PB2
;   PD4 †*。        。* ↓  PB1
;   PD5 †*。        。* ↓  PB0
;   GND   。10  11  。* ↓  PD6
.include "1200def.inc"       ; 必须写器件配置文件
    rjmpRESET                ;Reset Handle
.org  $005
RESET:
         LDI r16,0XFF        ;设 B 口、D 口为输出
         OUT ddrb,R16        ;设 b 口方向寄存器为输出
             OUT DDRD,R16    ;设 D 口方向寄存器为输出
             out portd,r16   ;关 D 口 LED,SL-AVR 实验器硬件设定高电平 LED 灯灭
             out portb,r16   ;关 B 口 LED
start:       ldi R17,0x08    ;循环次数
             ldi r18,0x7f    ;0b0111 1111,SL-AVR 实验器硬件设定低电平 LED 灯亮
loop:        out portb,r18   ;B 口.7 位 LED 灯亮
             sec         ;c=1
             ror r18     ;通过进位右循环
             rcall  delay    ;调用延时子程序
             dec r17     ;-1
             brne loop       ;检测 R17 循环不 0 为转移,为 0 按顺序执行
             out portb,r16   ;关 B 口
             ldi r18,0xbf    ; 0b1011 1111
             out portd,r18   ;D 口.6 位 LED 灯亮
             rcall  delay    ;延时
             ldi r18,0xff
             out portd,r18   ;关 D 口
             rjmp  start;循环
delay:       ldi r29,0x0a    ;延时子程序
delay1:      dec r30         ;复位后 R30=0X00
```

```
        brne delay1 ;R30 不为 0 转,为 0 按顺序执行
        dec r31     ;复位后 R31=0X00
        brne delay1 ;R30 不为 0 转,为 0 按顺序执行
        dec r29     ;复位后 R29=0X00
        brne delay1 ;R29 不为 0 转,为 0 按顺序执行
        ret         ;子程序返回
```

(2) SLAVR722.ASM :测试验证 AVRDIP40 引脚输出和 SL-AVR 开发下载实验器功能

```
.include "8515def.inc"   ;必须写器件配置文件
.org  $0000
    rjmpRESET            ;Reset Handle
.deftemp=r20
.defzh  =r31
.org $0010
RESET:
    ldi r16,high(RAMEND) ;设堆栈
    out SPH,r16           ;
    ldi r16,low(RAMEND)
    out SPL,r16
    ser temp             ;直接装入$FF,
    out DDRA,   temp     ;口的方向寄存器设定,为输出
    out DDRB,   temp
    out DDRC,   temp
    out DDRD,   temp
forever:
    clr temp             ;硬件设低电平 LED 灯亮
    out PORTA,  temp     ;PORTA 口 LED 灯亮
    out PORTB,  temp     ;B 口 LED 灯亮
    out PORTC,  temp     ;C 口 LED 灯亮
    out PORTD,  temp     ;D 口 LED 灯亮
    ldi R16,0X56         ;装延时常数,灯亮延时 1 秒,可修改该参数,应另存一个文件名
    rcall   delay        ;调用延时子程序
    ser temp             ;硬件设高电平 LED 灯灭
    out PORTA,  temp     ;PORTA 口 LED 灯灭
    out PORTB,  temp     ;B 口 LED 灯灭
    out PORTC,  temp     ;C 口 LED 灯灭
    out PORTD,  temp     ;D 口 LED 灯灭
    ldi R16,0X48         ;装延时常数,灯灭延时 0.5 秒,可修改该参数
    rcall   delay        ;调用延时子程序
    rjmpforever          ;无限循环
delay:                   ;通用延时子程序略,R16=$56,延时 1 秒,$67 延时 2 秒,
......
```

(3) SLAVR723.ASM ： 测试验证 AVRDIP40 引脚输出和 SL-AVR 开发下载实验器功能

测试 A 口、B 口、C 口、D 口 LED 灯亮循环变速移位

```
;   DIP40 AT90S8515 引脚排列图,"*"表示引脚上接 LED 灯
;              "↓↑"表示 LED 亮灯移动方向
;
;   PB0  ↓* 。140 。VCC
;   PB1  ↓* 。    。* ↑ PA0
;   PB2  ↓* 。    。* ↑ PA1
;   PB3  ↓* 。    。* ↑ PA2
;   PB4  ↓* 。    。* ↑ PA3
;   PB5  ↓* 。    。* ↑ PA4
;   PB6  ↓* 。    。* ↑ PA5
;   PB7  ↓* 。    。* ↑ PA6
;  /RESET   。    。* ↑ PA7
;   PD0  ↓* 。    。    ICP
;   PD1  ↓* 。    。    ALE
;   PD2  ↓* 。    。    OC1B
;   PD3  ↓* 。    。* ↑ PC7
;   PD4  ↓* 。    。* ↑ PC6
;   PD5  ↓* 。    。* ↑ PC5
;   PD6  ↓* 。    。* ↑ PC4
;   PD7  ↓* 。    。* ↑ PC3
;   XTAL2    。    。* ↑ PC2
;   XTAL1    。    。* ↑ PC1
;    GND     。20   21 。* ↓ PC0

.include "8515def.inc"      ; 器件配置文件
rjmp    RESET               ;Reset Handle
.org  $00d                  ;跳过中断区
RESET:      LDI R16,$5F     ;必须先设堆栈,因为复位后 SPL=0X00,SPH=0X00
            OUT SPL,R16     ;AVR 进堆栈是-1,出栈时+1,与 MCS-51 进出栈方向相反
            LDI R16,$02     ;
            OUT SPH,R16     ;设堆栈底为$025F,为 AVR AT90S8515
                            ;内部 SRAM($0060-$025F)底
        LDI r16,0XFF        ;
        OUT DDRB,R16        ;设方向寄存器为输出
            OUT DDRD,R16
            out ddra,r16
            out ddrc,r16
            out portd,r16   ;关 D 口,硬件设定高电平 LED 关
            out portb,r16   ;关 B 口,硬件设定高电平 LED 关
            out porta,r16   ;关 A 口,硬件设定高电平 LED 关
```

```
            out portc,r16   ;关 C 口,硬件设定高电平 LED 关
st:         ldi r28,0x08    ;循环次数
startb:     ldi R17,0x08
            ldi r18,0xfe    ;0b1111 1110
loopb:      out portb,r18   ;开 b 口.0 位 LED 灯亮,如何修改使 2 个
                            ;或 3 个或 1 隔 1 等 LED 灯亮移位
            sec             ;置进位标志 C=1
            rol r18         ;通过进位左循环
            mov r29,r28     ;移位(延时)次数
            rcall  delay    ;调用延时子程序
            dec r17     ;
            brne loopb      ;R17 不为 0 转,为 0 顺执
            out portb,r16   ;关 B 口
startd:     ldi R17,0x08
            ldi r18,0xfe    ;0b1111 1110
loopd:      out portd,r18   ;开 d 口.0 位 LED 灯亮
            sec     ;C=1
            rol r18         ;通过进位左循环
            mov r29,r28
            rcall  delay
            dec r17
            brne loopd
            out portd,r16
startc:     ldi R17,0x08
            ldi r18,0xfe    ;0b1111 1110
loopc:      out portc,r18   ;开 c 口.0 位 LED 灯亮
            sec
            rol r18         ;通过进位右循环
            mov r29,r28
            rcall  delay
            dec r17
            brne loopc
            out portc,r16
starta:     ldi R17,0x08
            ldi r18,0x7f    ;0b0111 1111
loopa:      out porta,r18   ;开 a 口.7 位 LED 灯亮
            sec
            ror r18         ;通过进位左循环
            mov r29,r28
            rcall  delay
            dec r17
            brne loopa
            out porta,r16   ;关 a 口
            dec r28
```

```
            breq  st        ;r28 为 0 转
            rjmp  startb    ;循环
delay:      ldi r31,0x23    ;延时子程序,可修改时间常数
delay1:     dec r30
            brne delay1
            dec r31
            brne delay1
            dec r29         ;移位速度次数
            brne delay
             ret            ;子程序返回
```

(4) SLAVR724.ASM：调节延时时间,就可用 AVR 的 I/O 口发出 1234567 音符声,

```
.include "8515def.inc"      ;器件配置文件
.org  $0000
    rjmpRESET
.org $0010
RESET:  ldi    r16,0x02
        out    sph,r16
        ldi    r16,0x5f
        out    spl,r16      ;设堆栈为 0X025F
        ldi     r16,0xff    ;设口为输出状态
        out     ddra,r16
        out     ddrb,r16
        out     ddrc,r16
        out     ddrd,r16
    out     porta,r16       ;关口,灭 LED 灯
        out     portb,r16
    out     portc,r16
        out     portd,r16
        ldi     r18,0x20    ;设延时常数
        ldi     r17,0x01
        ldi     r19,0x60
loop:   mov     r16,r19
        rcall   delay       ;调用延时子程序
        eor     r18,r17     ;异或
        out     portc,r18   ;输出 AT90S8515 的 C 口引脚
        dec     r20         ;-1
        brne    loop        ;R20 不为 0 转,为 0 顺执
        subi    r19,0x05    ;R19 减立即数
        cpi     r19,0x1f    ;R19 与立即数比
        brne    loop        ;R19 不 0 为转
        RJMP    RESET       ;循环
```

```
delay:  push    r16             ;2t 延时子程序
delay1: dec     r16             ;1t
        brne    delay1          ;1t/2t
        pop     r16             ;2t
        dec     r16             ;1t
        brne    delay           ;1t/2t
        ret                     ;4t
```

(5) SLAVR725.ASM:利用延时程序 I/O 口输出报警声

```
.include"8515def.inc"    ; 器件配置文件
.org    $0000
reset:ldi r16,$5f        ;设堆栈
        out  spl,r16
        ldi  r16,$02
        out  sph,r16
        ldi  r18,0xff    ;设口为输出
        out  ddrc,r18
      ldi r19,0xf0       ;报警参数
  lp: sbi  portc,$00     ;开 pc 口
        rcall  delay     ;延时
        cbi  portc,$00   ;关 pc 口
        rcall  delay
      dec r19            ;-1
      brne lp            ;r19 不为 0 转,为 0 顺执
      rcall delay1       ;较长延时,不发声
         rjmp lp         ;循环报警
delay1:ldi r17,$40       ;延时子程序,报警声快慢调节$30-$60
      rcall delay0
      ret
delay:ldi    r17,$9      ;延时子程序,报警声频率可调$a-$7
      rcall delay0
            ret
delay0:                  ;通用延时子程序略
.....
```

(6) SLAVR726.ASM: AT90S8515 的 PA 口使用建表方式的 LED 广告灯演示程序,

```
.include"8515def.inc"        ;器件配置文件
.org     $0000                ;设置起始地址
.equ     leddata=0x0250
   rjmp     reset
.cseg
.org     $0010
RESET:ldi    r16,$5f                  ;设置堆栈
         out    spl,r16
```

```
        ldi    r16,$02
        out    sph,r16
        ldi    r16,$90
        mov    r15,r16
        ser    r16                    ;设置 A 口为输出口
        out    ddra,r16               ;设置 A 口方向寄存器
    L0:ldi    zl,low(leddata*2)
        ldi    zh,high(leddata*2)
    L1:lpm
        mov    r16,r0
        cpi    r16,$0a
        breq   L0
        out    porta,r16
        rcall  delay                  ;调用延时子程序
        ld     r0,z+
        rjmp   L1
DELAY:                                ;通用延时子程序从略

.cseg                                 设置 LED 广告灯数据表
.org   leddata
.db 0xfe,0xfd,0xfb,0xf7,0xef,0xdf,0xbf,0x7f,0xbf,0xdf,0xef,0xf7,0xfb,0xfd
.db 0xfe,0xfd,0xfb,0xf7,0xef,0xdf,0xbf,0x7f,0xbf,0xdf,0xef,0xf7,0xfb,0xfd
.db 0x00,0x18,0x3c,0x7e,0xff,0x7e,0x3c,0x18
.db 0x00,0x18,0x3c,0x7e,0xff,0x7e,0x3c,0x18
.db 0xf8,0xf1,0xe3,0xc7,0x8f,0x1f,0x8f,0xc7,0xe3,0xf1
.db 0xf8,0xf1,0xe3,0xc7,0x8f,0x1f,0x8f,0xc7,0xe3,0xf1
.db 0xfe,0xfc,0xf8,0xf0,0xe0,0xc0,0x80,0x00,0x80,0xc0,0xe0,0xf0,0xf8,0xfc
.db 0xfe,0xfc,0xf8,0xf0,0xe0,0xc0,0x80,0x00,0x80,0xc0,0xe0,0xf0,0xf8,0xfc
.db 0xff,0xe7,0xdb,0xbd,0x7e,0xbd,0xdb,0xe7
.db 0xff,0xe7,0xdb,0xbd,0x7e,0xbd,0xdb,0xe7
.db 0xff,0x00
.db 0xff,0x00
.db 0xff,0x00
.db 0xff,0x00
.db  0x0a,0x0a
```

(7) SLAVR727.ASM : LED 发光二极管加 1 计数程序

```
;AT90S8515 的 PB、PD 口设计成十六位二进制加 1 计数程序,用 LED 发光二极管显示
.include"8515def.inc"
.org    $0000                 ;设置起始地址
    AB:ldi   r16,$5f          ;设置堆栈
       out   spl,r16
       ldi   r16,$02
```

```
      out  sph,r16
RESET:ldi  r18,0xff       ;设置 B 口,D 口为输出口
      out  ddrb,r18       ;设置 B 口,D 口方向寄存器
      out  ddrd,r18
      clr  r0
      clr  r1
   L0:mov  r2,r0
      mov  r3,r1
      com  r2             ;R2,R3 取反
      com  r3
      out  portb,r2       ;R2,R3 数据送 B 口,D 口
      out  portd,r3
      rcall delay         ;调用延时子程序
      inc  r0             ;R0 加 1,不为 0 跳转,为 0 顺执
      brne L0
      inc  r1
      brne L0
      dec  r0
      dec  r1
   L1:mov  r2,r0
      mov  r3,r1
      com  r2             ;R2,R3 取反
      com  r3
      out  portb,r2       ;R2,R3 数据送 B 口,D 口
      out  portd,r3
      rcall delay         ;调用延时子程序
      dec  r0             ;R0 减 1,不为 0 跳转,为 0 顺执
      brne L1
      dec  r1
      brne L1
      rjmp reset
DELAY:ldi  r18,$01         ;延时子程序
   L2:dec  r16
      brne L2
      dec  r17
      brne L2
      dec  r18
      brne L2
      ret
```

7.3　综合程序

7.3.A　LED/LCE/键盘扫描综合程序源程序

源程序：SLAVR73A.ASM

综合程序功能如下：

(1)．LED 及 LCD 显示程序,有自动识别 LED 或 LCD 功能,设 LCD 优先级高;

(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 上尖头字符左右移位程序

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

7.3. B　LED 键盘扫描综合程序

源程序：SLAVR73B.ASM

综合程序功能如下：

(1)．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;

(2) 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;

(3) 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

程序清单见光盘文件 SLAVR73B.ASM

7.3.1 在 SL-AVR 开发实验器 LED 上实现字符 8 的循环移位显示程序

```
; 源程序: SLAVR731.ASM;本程序在 SL-AVR 开发实验器上通过
;请你 1.如何修改字形; 2.改变字符个数,二位或三位或一隔一显示;
;3. 改变字形移动方向; 4.改变字符移位速度;
;
.include"8515def.inc"          ;器件配置文件
.def temp=r16                  ;数据暂存器
.def scndp=r22                 ;LED 显示位置暂存器
```

h	g	f	e	d	c	b	a	十六进制 码	字形
PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0		
0	0	1	1	1	1	1	1	3F	0
0	0	0	0	0	1	1	0	06	1
0	1	0	1	1	0	1	1	5B	2
0	1	0	0	1	1	1	1	4F	3
0	1	1	0	0	1	1	0	66	4
0	1	1	0	1	1	0	1	6D	5
0	1	1	1	1	1	0	1	7D	6
0	0	0	0	0	1	1	1	07	7
0	1	1	1	1	1	1	1	7F	8
0	1	1	0	1	1	1	1	6F	9
0	1	1	1	0	1	1	1	77	A
0	1	1	1	1	1	0	0	7C	B
0	0	1	1	1	0	0	1	39	C
0	1	0	1	1	1	1	0	5E	D
0	1	1	1	1	0	0	1	79	E
0	1	1	1	0	0	0	1	71	F
1	1	1	1	0	0	1	1	F3	P.
0	1	1	1	0	1	1	0	76	H

硬件设定高电平
LED笔划点亮
低电平LED管选中

```
.org $0000
    rjmp reset
.org $030
reset:  ldi temp,low(ramend);设置堆栈指针。
    out spl,temp
    ldi temp,high(ramend)
    out sph,temp
    ldi temp,$ff     ;设置 B、D 口输出。
    out ddrb,temp
    out ddrd,temp
    out portd,temp
    ldi temp,$7f     ;字形 8 的代码为$7F(可修改)。
    out portb,temp
again:  sec          ;置进位位为 1(低电平 LED 亮,高电平 LED 灭)
    ldi scndp,0b11011111 ;扫描显示 SCANDP(可修改)
route:  out portd,scndp      ;从 LED 最左一位(D5)右移(可修改)
    ldi temp,$40     ;设置延时常数(可修改)。
    rcall delay      ;调用延时
    ror scndp        ;右循环(可修改)
```

```
    brcc again      ;显示下一位
    rjmp route      ;循环显示
delay:              ;通用延时子程序略。
```

7.3.2 电脑放音机

源程序：SLAVR732.ASM

AVR 单片机在儿童智能玩具中的应用--音乐玩具

利用单片机开发儿童智力玩具大有作为,尤其单片机扩展存贮器方便，而大容量存贮器价格也很低,64KB 的 EPROM 可存放 300 多首歌曲,8M 位 EPROM 可存放 5000 多首歌曲,几个芯片就可组成一个音乐库,这是用其它方法难办的。

利用 AVR 单片机产生乐曲音符,再把乐谱翻译成计算机音乐语言，由单片机进行信息处理,再经过信号放大,由耳机或喇叭放出乐曲声．由于音符和节拍是由计算机产生的,所以发音音符和节拍准确,可见音乐从娃娃开始抓起，音乐玩具是儿童第一个好老师。利用单片机的中断,I/O 口控制功能，可以做到电脑放音机有自动连续放音功能,乐曲全部放完自动从头开始连续放音,循环不断。

如何产生音乐频率:

1.要产生音频脉冲,只要算出某一音频的周期(1/频率),然后将此周期除以 2，即为半周期的时间,然后利用计时器计时此半周期时间,每当计时到后就将输出脉冲对 I/O 口反相,然后重复计时此半周期时间再对 I/O 口反相,如此就可在 I/O 口引脚上得到此频率的脉冲(程序驱动 I/O 口反相,即正、负各半周期为一个周期,才能使喇叭"吸、放"发声);

2．利用 AVR 单片机的内部计时器让其工作在计数模式 MODE1(16 位定时计数器)下,改变计数值 TCNT1H 及 TCNT1L 以产生不同的频率;

3．例如以 6MHZ 晶振为例:要产生频率为 523HZ,其周期 T= 1/523= 1912us,其半周期为 1912/2=956us,因此只要令计数器计时 956us/1us=956(为半周期).所以在每计数 956 次时将 I/O 反相,就可得到中音 DO(523HZ).

计数脉冲值与频率的关系公式如下:

N=Fi(6MHz 晶振,CPU 产生的频率)÷2(半周期)÷Fr

N:计数值

Fi:以 6MHZ 晶振为例,内部计时(数)一次需 2us，频率单位为 1 周期/秒,即 HZ

1 周期/2us=1 周期/2X10^{-6}秒=500000 次/秒=500000HZ

故其频率为 500000HZ

Fr:要产生的频率

4．其计数值的求法如下:

T(16 位计数器计多少后溢出)=65536(16 位二进制计数器,计满数溢出时的计数值为 2 的 16 次方)-N=65536-Fi/2/Fr

例如:求低音 DO(262HZ),中音 DO(523HZ),高音 DO(1046HZ) 的计数值?

设 K=65535 F=500000=Fi=0.5MHZ

T=65536-N=65536-Fi/2/Fr=65536-500000/2/Fr=65536-250000/Fr

低音 DO 的 T=65535-1908=63627(十进制数)

中音 DO 的 T=65535-0956=64579(十进制数)

高音 DO 的 T=65535-0478=65057(十进制数)

5．C 调各音符频率与计数值 T 的对照表：

音符	频率 HZ	半周期	TCNT 值	音符	频率 HZ	半周期	TCNT 值
低 1DO	262	1908μS	63627	#4FA#	740	0676μS	64859
#1DO#	277	1805	63730	中 5SO	784	0638	64897
低 2RE	294	1700	63835	#5SO#	831	0602	64933
#2RE#	311	1608	63927	中 6LA	880	0568	64967
低 3M	330	1516	64020	#6LA#	932	0536	64999
低 4FA	349	1433	64012	中 7SI	988	0506	65029
#4FA#	370	1350	64185	高 1DO	1046	0478	65057
低 5SO	392	1276	64259	#DO#	1109	0451	65084
#5SO#	415	1205	64330	高 2RE	1175	0426	65109
低 6LA	440	1136	64399	#2RE#	1245	0402	65133
#6LA#	466	1072	64463	高 3M	1318	0372	65156
低 7SI	494	1012	64523	高 4FA	1397	0358	65177
中 1DO	523	0956	64579	#4FA#	1480	0338	65197
#1DO#	554	0903	64632	高 5SO	1568	0319	65216
中 2RE	578	0842	64683	#5SO#	1661	0292	65243
#2RE#	622	0804	64731	高 6LA	1760	0284	65251
中 3M	659	0759	64776	#6LA#	1865	0268	65267
中 4FA	698	0716	64819	高 7SI	1976	0253	65282

"#"表示半音,用于上升或下降半个音

如何产生节拍：

每个音符使用 1 个字节，每个节拍使用 1 个字节,AVR 程序存储器可以设为 16 位,即 1 个字,或称双字节,所以一个字的高 8 位存放音符码,低 8 位存放节拍码。如果 1 拍节为 0.4 秒则 1/4 拍是 0.1 秒,只要设定延迟时间就可求得节拍的时间,我们假设 1/4 拍为 1 DELY 单位,则 1 拍应为 4 个 DELY,以此类推,只要求得 1/4 拍的 DELY 单位时间,其余的节拍就是它的倍数.

1/4 拍的延迟时间=187 毫秒

节拍与节拍码对照表

节拍码	节拍数(拍)	节拍码	节拍数(拍)
1	1/4	1	1/8
2	2/4	2	1/4
3	3/4	3	3/8
4	1	4	1/2
5	1 又 1/4	5	5/8
6	1 又 1/2	6	3/4
8	2	8	1
10	2 又 1/2	10	1 又 1/4
12	3	12	1 又 1/2
16	1 又 3/4		

建立音乐的步骤：

找出乐曲，然后对照音符表,翻译出乐曲码,用程序伪指令 DB 输入曲码和节拍码;也可直接

在调试窗口的程序存储器窗口$0100 地址输入曲码和节拍码(只适用于在线实时仿真器)。

例:音符表练习,

1.把简谱翻译成曲码代码;

以下音符均设为一拍,代码为 4

1 2 3 4 5 6 7(低八度音)1 2 3 4 5 6 7 (中音) 1(高音) 1(高音) 7 6 5 4 3 2 1(中音) 7 6 5 4 3 2 1(低八度音)

曲码	1	3	5	6	8	10	12	13	15	17	18	20	22	24	25
简码	1	2	3	4	5	6	7	1	2	3	4	5	6	7	1
	低八度音							中 音							高音

曲码	36	34	32	30	29	27	25	24	22	20	18	17	15	13	12
简码	7	6	5	4	3	2	1	7	6	5	4	3	2	1	7
	高八度音							中 音							低音

最后翻译成乐曲加节拍代码为:

01,04,03,04,05,04,06,04,08,04.10,04,12,04,13,04,13,04,15,04,17,04,18,04,20,04,
22,04,24,04,25,04,
25,04,36,04,34,04,32,04,30,04,30,04,29,04,27,04,25,04,24,04,22,04,20,04,18,04,
17,04,15,04,13,04,12,04

以上乐曲数据用伪指令 DB 方式输入"乐曲.ASM"的$0100 地址,再汇编一次就可下载试听,

注意:音符节拍间用逗号隔开,不要不小心键入小数点,因为逗号键右边是小数点键,

键入小数点,程序汇编时将造成计算机死机!

00 00(4 个零为所有曲结束标志)

1. 把乐曲代码输入计算机

把 SL-AVR 实验器与 PC 机连机,U4 插上 AT90S8515 芯片,插上音响器短路块,开机通电。进入 AVR 下载窗口,进行下载操作,下载结束应能听到乐曲声。

```
;************** 乐曲程序 SLAVR732.ASM ************************
;* 标题:AT90S8515 C 口输出乐曲声—电脑放音机
;* 版本:        1.0
;* 最后更新日期: 2000.08.08
;* 支援  E-mail:     gzsl@sl.com.cn
;* 描述
;*   用 SL-AVR 万用下载开发实验器做样机,在 AT90S8515 的 C 口接喇叭发出乐曲声,
;* 请你把最喜爱的乐曲送入单片机! 起始地址为$0100,也可把曲码、节拍码在调试窗口中的
;* 程序存储器窗口(Program    Memory)内,从$0100 地址,
;*   用键盘直接输入乐曲(仅适合 ICE-200 实时仿真器)
;* 作者: SL.
;*程序适用于所有单片机
;*******************************************************************************

.include"8515def.inc"          ;文件头 AT90S8515 器件配置文件,不同的器件有不同的器件配置文件
      rjmp RESET               ;AVR 重新定位
.def  TEMPDH      =r2          ;寄存器定义
.def  TEMPDL =r3
.def  CNT =r10
```

```
.def  SCNN    =r11
.def  KEYN    =r12
.def  SCNK    =r13
.def  SCNDP   =r14
.def  KSNI    =r15
.def  TEMP    =r16          ;数据暂存器
.def  TEMP1   =r17
.def  TEMP2   =r18
.def  TEMP3   =r19
.def  SCNTT   =r26;
.def  MUSN    =r22          ;输出乐曲声暂存器
.def  TONL    =r21          ;节拍码低位
.def  TONH    =r20          ;节拍码高位
.def  PLYTON =r25           ;存乐曲码
.def  TONSET =r24
.def  TONLNG      =r23      ;存节拍码
.cseg
.org 0x06                   ;TIM1__OVF 定时器 1 溢出中断处理入口地址
intt1:      RJMP    OUTPM   ;转定时器 1 溢出中断处理,发音周期到,则跳转到发音输出态
.cseg
.org 0x010                  ; 定时器 1 溢出中断处理程序,发音起始地址
                            ;发音周期到重新装入计时值并将输出到 PORTC 口
OUTPM: OUT      TCNT1H,TONH ;重新将 TONH 新计时值载入 TCNT1H 内
       OUT      TCNT1L,TONL ; 重新将 TONL 新计时值载入 TCNT1L 内
       SBIS     PORTC,00    ;先检测 PORTC 口是否为 1 而跳转
       RJMP     SETOP1      ;若是 PORTC 口为 0 则跳到 SETOP1 令 PORTC 口转为 1
SETOP0: CBI  PORTC,00       ;若 PORTC 为 1 则令 PORTC 转为 0
       LDI  MUSN,$00        ;同时令 MUSN 为 00 值
       RETI                 ;回中断前主程序并令可再次中断返回
SETOP1: SBI  PORTC,00       ;若 PORTC 为 0 则令 PORTC 转为 1
       LDI  MUSN,$01        ;同时令 MUSN 为 01 值
       RETI                 ; 回中断前主程序并令可再次中断返回
.cseg
.org 0x020                  ;主程序起始地址,必须跳过中断区
RESET:
       ldi    temp,low(RAMEND)      ;RAMEND 为 8515def.inc 内建值为 025FH
       out    SPL,temp              ;启始堆栈指针低位将 TEMP=02H 放入 SP=3DH
;若硬件堆栈或者片 AVR 片内含 SRAM 小于 256B 时,下列二行程序可省略,
       ldi    temp,high(RAMEND)     ;以 TEMP=R16<5F 为数据装入缓冲暂存器
       out    SPL+1,temp            ;堆栈指针高位将 R16=TEMP=5FH 放入 SPL+1=3EH
       wdr                          ;在使用看门狗计时器前需重设看门狗计时器,为
                                    ;避免在接下来程序前就因 WDT 已快计时溢出而重设
       ldi    temp,$0F              ;WDTCR 地址$21 设定以 TEMP 缓冲令 WDE=D3=1
       out    WDTCR,temp            ;并会预除为 2048mS 设定 WDE=1=D3 输出到 WDTCR 内
```

```
        LDI  MUSN,$00                ;令 MUSN 为 00 值
        ldi    temp,$00              ;令 TEMP 暂存器放入 00
        OUT      TCCR1A,TEMP         ;TEMP=00 内含输出到 TCCR1A 内禁止比较器及 PWM 动作
        OUT      TCCR1B,TEMP         ;将 TEMP=00 内含输出到 TCCR1B 内停止 TC1 计时及捕抓
        LDI      TEMP,$02            ;将 02 值预存入 SRAM 的 0100H 内作 TC1 的
        STS      $0100,TEMP          ;TCCR1B 控制内含令 TC1 为计时预除 8
LEDA:   CLI                          ;中断总开关 sreg=d7=i=1
        Ldi      r16,0b10000000          ;令 toiel=1 触发中断
        out      timsk,r16               ;将 R16 的 D7=1 令 TOIE1=1 触发中断
        LDI      TEMP,$FF            ;设 AVR 单片机 I/O 口方向宏寄存器为输出
        OUT      DDRA,TEMP           ;A 口为输出
        OUT      DDRB,TEMP           ;B 口为输出
        OUT      DDRC,TEMP           ;C 口为输出
;       OUT      DDRD,TEMP           ;D 口为输出
        LDI      TEMP,0B11111111     ;关灭 I/O 口的 LED 发光二极管
        OUT      PORTC,TEMP          ;C 口输出乐曲声
        OUT      PORTA,TEMP          ;关 A 口 LED 灯,硬件设定高电平 LED 暗
        OUT      PORTB,TEMP          ;关 B 口 LED 灯,硬件设定高电平 LED 暗
        OUT      PORTD,TEMP          ;关 D 口 LED 灯,硬件设定高电平 LED 暗
        CLR      TEMP2               ;暂存器清零
        CLR      TEMP1               ;暂存器清零
        CLR      KSNI                ;暂存器清零
        LDI      SCNTT,$02           ;
        CLR      TONLNG              ;暂存器清零
STARTP: WDR                          ;关看门狗
        LDI      ZH,HIGH(PLYTAB*2)   ;启动演奏则令数据装入 Z 地址
        LDI      ZL,LOW(PLYTAB*2)    ;音乐演奏乐谱存放在 PLYTAB*2 起始地址
NEXMUT:   LPM                    ;将 Z 所指程序存储器乐曲,依次取音符码及节拍码置于 R0
        MOV      PLYTON,R0       ;将取出的第一个音符码装入 PLYTON 作周期控制
        LD       R0,Z+           ;以 LD R0,Z+指令使得 Z 间接寻址加 1
        LPM                      ;将 Z 所指程序存储器乐曲,依次取节指码置于 R0
        MOV      TONLNG,R0       ;将取出的第一个节拍码装入 PLYTON 作节拍控制
        OR       R0,PLYTON       ;将此 R0 节拍码与音符码 PLYTON 作 OR 运算
        LD       R0,Z+           ;以 LD R0,Z+指令使得 Z 间接寻址加 1
        BRNE     PLAYM           ;若音符码及节拍码非为全 00 值,则跳到 PLAYM 演奏
        LDI      TEMP,$00        ;若音符码及节拍码全为零(0000),则为乐曲结束标记
        OUT      TCCR1B,TEMP     ;将 TEMP=00 内含输出到 TCCR1B 内停止 TC1 计时及捕抓
        CLI                      ;令中断总开关 SREG 的 I 标志清零,而禁止中断
        SBI      PORTD,00        ;令 PINC=1 将喇叭输出 OFF
        RJMP     STARTP          ;循环演奏
PLAYM:  PUSH     ZH              ;进栈保存数据
        PUSH     ZL              ;进栈保存数据
        TST      PLYTON          ;检测 PLYTON 是否为 0
        BREQ     MUSTD           ;若为 0 则跳至 MUSTD 作节拍等待
```

```
    LDI  ZH,HIGH(MUSTAB*2)  ;乐曲码装入 Z 地址
    LDI  ZL,LOW(MUSTAB*2)   ;计时器值存放于 MUSTAB*2 起始地址
    MOV     TEMP,PLYTON     ;将乐曲码 PLYTON 装入 TEMP 寄存器内
    DEC TEMP                ;寄存器 TEMP 减 1
    LSL     TEMP            ;寄存器 TEMP 左移即 X2
    ADD     ZL,TEMP         ;将正确的计时器控制乐曲码的存表位移且使 TEMP 加入 ZL
    LDI     TEMP,$00        ;令 TEMP=00 以便让 ZH 与进行标志位 C 相加
    ADC     ZH,TEMP         ;将 ZH 与 TEMP=00 以及进行标志位 C 相加得到真正的 Z 地址值
    LPM                     ;将 Z 所指 PROM 的预存乐曲码计时长度低位值装入 R0
    MOV     TONL,R0         ;将乐曲码计时长度低位值 R0 装入 TONL 内
    OUT     TCNT1L,R0       ;将乐曲码计时长度低位值 R0 也装入 TCNT1L 内
    LD      R0,Z+           ;以 LD R0，Z＋指令使 Z 间接寻址加 1
    LPM                     ;将 Z 所指 PROM 之预存乐曲码之计时节拍码置于 R0
    MOV     TONH,R0         ;将节拍高位值 R0 装入 TONH 内
    OUT     TCNT1H,R0       ;  将节拍高位值 R0 装入 TCNT1H 内
    POP     ZL              ;出栈将 ZL,ZH 由堆栈指针依次取回
    POP     ZH              ;出栈
    LDS     TEMP,$0100      ;将 SRAM 地址 0100H 之内容装入 TEMP 内
    OUT     TCCR1B,TEMP     ;将原 0100H 的 SRAM 内容输出到 TCCR1B 控制 TC1
    SEI                     ;内容 02 令 TC1 为计时预除 8,令中断总开关 I=1 触发
MUSTD: RCALL    PLYDEL      ;调用延时子程序 0.2S
    DEC TONLNG              ;将节拍码 TONLNG 减 1
    BRNE    MUSTD           ;若节拍码 TONLNG 不为 0 则转回,再发音,为 0 则顺执
    RJMP    NEXMUT          ;继续到 NEXMUT 取乐曲码和节拍码
PLYDEL:LDI      TEMP,185    ;延时子程序,185x1mS=185mS,
                            ;即 PLYDEL=185mS 为十进制时间常数
DT3:    LDI     TEMP1,04    ;  送时间常数 4X250μS=1mS,DT3 约为 1mS
DT2:    LDI     TEMP2,250   ;250 为十进制时间常数,250X8X125nS=250mS,dt2=250μS
DT1:    WDR                 ;1T
        WDR                 ;2T
        WDR                 ;3T
        WDR                 ;4T
        WDR                 ;5T
        DEC TEMP2           ;6T,TEMP-1
        BRNE    DT1         ;8T,TEMP2 不为 0(则共执行 250X8XT=250μS)转,
                            ;为 0 按顺序执行
    DEC TEMP1               ;TEMP1-1
    BRNE    DT2             ;TEMP1  不为 0(则共执行 250X4US=1mS)转,为 0 按顺序执行
    DEC TEMP                ;TEMP-1
    BRNE    DT3             ;TEMP  不为 0(则共执行 185X1mS=185mS)转,为 0 按顺序执行
    RET                     ;子程序返回

    ;约定:因为计算机不能表示简谱乐曲,低音用数字后一点表示,
    ;高音用数字前一点表示,
```

```
;半音用#号,'为隔开音符,
;乐曲节拍应对照简谱查看,音长为节拍,
;一拍为 04,3/4 拍为 03,1/2(2/4)拍为 02,1/4 拍为 01,
;00 为表示休止符,
;00 00 连续两个字节为零,表示乐曲结束
```

;　　乐曲低八度音

曲码号	1	2	3	4	5	6	7	8	9	10	11	12
音符号	1	#1	2	#2	3	4	#4	5	#5	6	#6	7

;**********

;　　乐曲中音

曲码号	13	14	15	16	17	18	19	20	21	22	23	24
音符代	1	#1	2	#2	3	4	#4	5	#5	6	#6	7

;

;**********

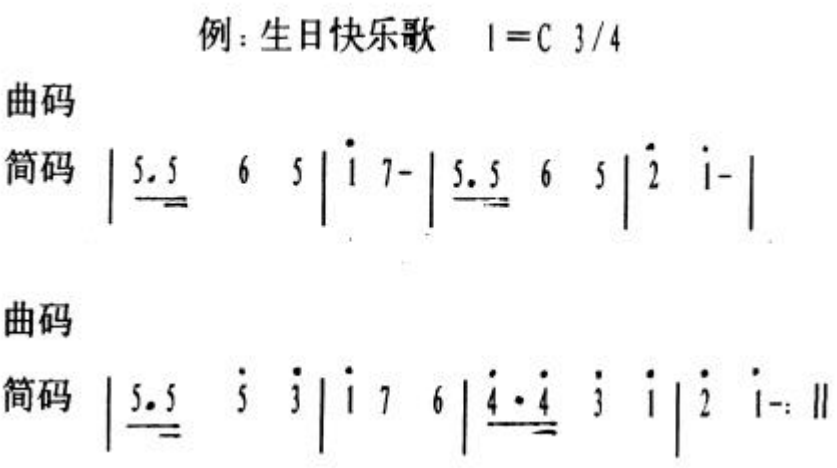
;　　乐曲高八度音

曲码号	25	26	27	28	29	30	31	32	33	34	35	36
音符号	1	#1	2	#2	3	4	#4	5	#5	6	#6	7

;

;**********

```
.EQU    PLYTAB=0X0100           ;乐曲存放首地址
.EQU    MUSTAB=0X00A0           ;乐曲音符表存放首地址
.cseg                           ;实例
.org PLYTAB     ;"祝你生日快乐" 乐曲 1=C 3/4  乐曲存放起始地址,请查看对照简谱乐曲
;
```




```
;
; 注意：曲码中不能用小数点,只能用逗号隔开,否则汇编时造成死机
    ;|   5              5,      6       5|
    .DB 20,02,00,01,20,01,22,04,20,04;
    ;|  .1       7       -|
```

```
    .DB 25,04,24,04,00,04
    ;|    5                5,        6        5|
    .DB 20,02,00,01,20,01,22,04,20,04
    ;|   .2       .1        -|
    .DB 27,04,25,04,00,04
    ;|    5                5,     .5       .3|
    .DB 20,02,00,01,20,01,32,04,29,04
    ;|   .1        7        6|
    .DB 25,04,24,04,22,04
    ;|   .4                .4       .3       .1|
    .DB 30,02,00,01,30,01,29,04,25,04
    ;|   .2       .1        -  |
    .DB 27,04,25,04,00,04
    ;REAGAIN
    ;|    5                5,        6        5|
    .DB 20,02,00,01,20,01,22,04,20,04
    ;|.1          7        -  |
    .DB 25,04,24,04,00,04
    ;|    5                5,        6        5|
    .DB 20,02,00,01,20,01,22,04,20,04
    ;|   .2       .1        -  |
    .DB 27,04,25,04,00,04
    ;|    5                5,     .5       .3|
    .DB 20,02,00,01,20,01,32,04,29,04
    ;|   .1        7        6|
    .DB 25,04,24,04,22,04
    ;|   .4                .4       .3       .1|
    .DB 30,02,00,01,30,01,29,04,25,04
    ;|   .2       .1        -  |
    .DB 27,04,25,04,00,04
;              万水千山总是情
.db 17,04,18,04,20,06,20,02,22,04,20,04,17,12,15,04  ;
.db 13,06,17,02,15,04,13,04,10,12,10,04,8,8,13,04    ;注意:08 应写成 8 才能编译通过
.db 15,04,17,04,20,04,22,04,17,04,15,15,15,04,00,04  ;
.db 17,04,18,04,20,06,20,02,22,04,20,04,17,12,15,04  ;
.db 13,06,17,02,15,04,13,04,10,12,10,04,8,8,13,06    ;
.db 17,02,15,06,13,02,13,04,10,04,13,15,13,8,17,04   ;
.db 20,04,22,12,25,10,22,04,18,04,20,06,22,02,20,12  ;
.db 17,04,20,8,17,04,20,04,22,12,25,04,25,04,22,04   ;
.db 20,04,17,04,15,15,15, 8,17,04,18,04,20,06,20,02  ;
.db 22,04,20,04,17,12,15,04,13,06,17,02,15,04,13,04  ;
.db 10,12,10,04,8,8,13,04,17,04,15,06,13,02,10,04
.db 12,04,13,15,13,15    ;
    .DB 00,00        ;END
```

```
.cseg
.org MUSTAB              ;音符表地址标号
    ;约定:低音为数字后一点表示,高音为数字前一点表示,
          ;  半音为#号,'为隔开音符

    ;1      2     3      4     5      6     7      8     9
    ;1.   '#1.   '2.   '#2.   '3.    '4.   '#4.    '5.   '#5.

    ;10   11   12   13    14   15   16   17   18
    ;6.   '#6.  '7.  '1    '#1  '2   '#2   '3   '4
    .DW 63627,63730,63835,63927
    .DW 64020,64102,64185,64259
    .DW 64330,64399,64463,64523
    .DW 64579,64632,64683,64731
    .DW 64776,64819

    ;19   20   21   22   23   24   25    26    27
    ;#4   '5   '#5  '6   '#6  '7    '  1   '#.1  '.2
    ;   28    29   30    31   32    33   34    35   36
    ;   '#.2  '.3  '.4   '#.4  '.5   '#.5  '.6  '#.6  '.7

    .DW 64859,64897
    .DW 64933,64967,64999,65029
    .DW 65057,65084,65109,65133
    .DW 65156,65177,65197,65216
    .DW 65243,65251,65267,65282
```

7.3.3 键盘扫描程序说明

源程序见 SLAVR73A(73B).ASM(**其中部分程序,键盘扫描程序,可供调用**)

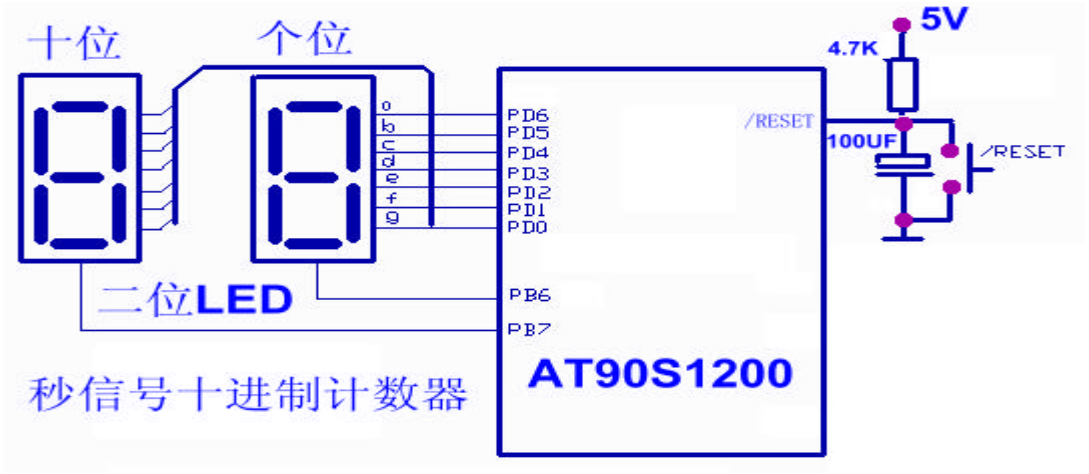
```
SCAN1:  PUSH XH          ;键扫显示子程序。
        PUSH XL          ;将 xl 压入堆栈
        PUSH TEMP3
        PUSH TEMP2
        PUSH TEMP1
        PUSH TEMP
        LDI XL,$60
            SET                     ;T 标志为 1 表示未按键
            LDI  SCNN,$00           ;按键起始扫描码 SCNN 为 00
            LDI  SCNDP,0B11011111   ;令 6 位七段 LED 扫描显示码初始为 11011111
            LDI  CNT,$06            ;七段 LED 共 6 位故 CNT=6 为位数计数
            LDI  KSNI,0B11110111    ;4*4 键盘扫描码 KSNI 初始为 11110111
      COL1: LDI  TEMP,$FF           ;PORTB 设定为输出
            OUT  DDRb,TEMP
```

```
    OUT   DDRC,TEMP       ;PORTC 设定为输出
OUT   PORTC,TEMP
    OUT   DDRd,TEMP       ;PORTD 设定为输出
OUT   PORTd,SCNDP         ;6 位七段 LED 扫描显示码输出到 PORTD
    LD    R1,X+      ;要显示于七段 LED 的间接寄存器 X 中的内容送入 R1 并令 X 加 1
    OUT   PORTb,R1            ;显示内容输出到 PORTB 以驱动 LED 显示
    RCALL   DELAY             ;调用延时以显示此位数一段时间
    MOV   TEMP,CNT            ;LED 位数为 6 而按键码行数为 4 故需作 CNT 值检测
    SUBI   TEMP,$03           ;CNT=TEMP 与 3 相减比较
    BRCS   NOSK               ;位数扫描 CNT 超过 3 则 C 为 1 跳到 NOSK 不作按键处理
    LDI   TEMP1,$04           ;一共要检查 4 个按键
    LDI   TEMP,0B00001111     ;设定 PC0-PC3 为输出 PC4-PC7 为输入
    OUT   DDRc,TEMP
    OUT   PORTc,KSNI          ;KSNI 输出到 PORTC 并令 PC7-PC4 为上拉电阻输入态
    RCALL   DELYT             ;调用延时以稳定读取键盘 I/O 输入端
    IN    TEMP,PINc           ;读取 C 口检测 PC7-PC4 看是否有按键低电位输入
    ANDI   TEMP,0B11110000    ;取 TEMP 的高 4 位
    SWAP   TEMP           ;键码顺序为 PC4-PC7 故将 TEMP 的高低 4 位互换成 D0-D3
  KROW: SEC          ;令 C 标志为 1 以便将键盘码 D0-D3 移到 C 标志位检测
ROR   TEMP        ;TEMP 的内容右移 1 位将第一个键码 D0=PC4 移到 C 标志位检测
    BRCS   NOKEY       ;若有键按下则测到 PC4=D0=0，若 C=1 无按键则转到 NOKEY
    CLT                ;若 PC4=D0=CF=0 表示有按键令 T=0 表示有按键
    MOV    KEYN,SCNN   ;把按键扫描码 SCNN 送键码 KEYN 中保存
SBIS PINd,$07
ADIW KEYN,$10          ;判定 SHIFT 键是否按下，按下则键值加 10
 NOKEY: INC   SCNN     ;按键扫描码 SCNN 加 1
    DEC   TEMP1        ;扫描读取键数 TEMP1 减 1
    BRNE   KROW        ;每行有 4 个按键如 TEMP1 不为 0 则跳到 KROW 再检测 PC5-PC7
    SEC       ;此行 4 个键码检测完后令 C 为 1 以方便键盘扫描码 KSNI 内容的移位
    ROR   KSNI ;键盘扫描码 KSNI=CF=1>11110111 移位以进行下一行按键扫描
  NOSK: SEC            ;令进位标志 CF=1
    ROR   SCNDP        ;将扫描显示码 SCNDP 左移作下一位扫描
    DEC   CNT          ;共需作 6 位数扫描显示故 CNT 减 1
    BRNE   COL1        ;CNT 减 1 不为 0 则跳回 COL1 再作扫描显示及读取键盘输入
    LDI    TEMP,$FF    ;若已完成全部扫描显示和读取按键则令 TEMP=0ff
    OUT    DDRC,TEMP   ;TEMP 输出到 DDRC 设定 PORTC 为输出驱动 LED
OUT    PORTC,TEMP
POP TEMP               ;出栈
POP TEMP1
POP TEMP2
POP TEMP3
POP XL
POP XH
RET                    ;子程序返回
```

7.3.4 十进制计数显示

源程序:SLAVR734.ASM
应用例子*.ASM,必须编译生成*.OBJ 文件才可调试,如要修改*.ASM,必须修改文件属性,去掉*.ASM 只读文件属性。
★ 必须按下图接线才能正常工作!

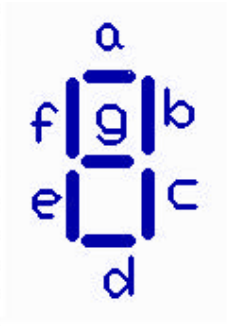
```
;*********************十进制计数程序 **********************************
;* 标题:用二位 LED 显示十进制计数
;* 版本:    1.0
;*最后更新日期: 2000.08.08
;* 支援 E-mail: gzsl@sl.com.cn
;* 描述
;* 用 AVR AT90S1200 的 D 口接二只 LED 数目管,PB7,PB6 作片选,硬件设定 D 口高电平 LED 灯
;* 亮,B 口低电平选中 LED,即选用共阴极数目管,最大显示十进制 99。硬件接线原理图如下:
;* 作者: SL.Z
;* 程序适用于所有单片机
;*****************************************************************************
```




```
.include"1200def.inc"    ;AT90S1200 配置文件
.org $0000
    rjmp reset
.org $0010
  reset:ldi r20,$ff      ;设 B 口、D 口为输出
        out ddrb,r20     ;B 口方向寄存器
        out ddrd,r20     ;D 口方向寄存器
        sbi $18,7        ;硬件设定 B 口低电平选中 LED,PB.7 关高位 LED,
        sbi $18,6        ;PB.6 关低位 LED
        ldi r20,$fe      ;建字形表
    mov r0,r20           ;0
        ldi r20,$b0
    mov r1,r20           ;1
```

```
    ldi r20,$ed
mov r2,r20          ;2
    ldi r20,$f9
```

h	a	b	c	d	e	f	g	十六进制码	字形	R
1	1	1	1	1	1	1	0	FEH	0	R0
1	0	1	1	0	0	0	0	B0H	1	R1
1	1	1	0	1	1	0	1	EDH	2	R2
1	1	1	1	1	0	0	1	F9H	3	R3
1	0	1	1	1	0	1	1	B3H	4	R4
1	1	0	1	1	0	1	1	DBH	5	R5
1	1	0	1	1	1	1	1	DFH	6	R6
1	1	1	1	0	0	0	0	F0H	7	R7
1	1	1	1	1	1	1	1	FFH	8	R8
1	1	1	1	0	0	1	1	F3H	9	R9
1	1	1	1	0	1	1	1	F7H	A	R10
1	0	0	1	1	1	1	1	9FH	B	R11
1	1	0	0	1	1	1	0	CEH	C	R12
1	0	1	1	1	1	0	1	BDH	D	R13
1	1	0	0	1	1	1	1	CFH	E	R14
1	1	0	0	0	1	1	1	C7H	F	R15



硬件设定高电平

LED数目管点亮

```
mov r3,r20          ;3
    ldi r20,$b3
mov r4,r20          ;4
    ldi r20,$db
mov r5,r20          ;5
    ldi r20,$df
mov r6,r20          ;6
    ldi r20,$f0
mov r7,r20          ;7
    ldi r20,$ff
mov r8,r20          ;8
    ldi r20,$f3
mov r9,r20          ;9
    ldi r20,$f7
mov r10,r20         ;A
    ldi r20,$9f
mov r11,r20         ;B
    ldi r20,$ce
mov r12,r20         ;C
    ldi r20,$bd
mov r13,r20         ;D
    ldi r20,$cf
mov r14,r20         ;E
    ldi r20,$c7
mov r15,r20         ;F
bclr 7              ;清 I 标志,关中断
```

```
    clr r28             ;(28)=$00
main:   ldi r20,$28     ;扫描次数
start:  mov r30,r28     ; 十位显示字符值,第一次显示 0
display:andi r30,$f0    ; 取十位
    swap r30        ;半字节交换,获取 Z 地址
ledh:   ld r25,z    ;LED 高位,复位后(R31)=$00
        out portd,r25   ;送 D 口显示
        sbi $18,6       ;关个位,硬件设定高电平不亮
        cbi $18,7       ;选通十位,硬件设定低电平亮
        ldi r27,$10     ;延时常数
delay1: dec r26         ;-1,延时
    brne delay1         ;不为 0 转
    dec r27             ; -1,延时
    brne delay1         ; 不为 0 转
        sbi $18,7       ; 关十位
        nop
    mov r30,r28         ;个位显示字符值, 第一次显示 0
    andi r30,$0f    ;取个位
ledl:   ld r25,z    ;荻显示字符
        out portd,r25   ;送 D 口显示
        sbi $18,7       ; 关十位,硬件设定高电平不亮
        cbi $18,6       ; 选通个位,硬件设定低电平亮
        ldi r27,$10     ;延时常数
delay2: dec r26         ; -1,延时
        brne delay2     ; 不为 0 转
    dec r27             ; -1,延时
        brne delay2     ; 不为 0 转
        sbi $18,6       ; 关个位
    nop
        dec r20         ;显示数-1
        brne start      ;(R20)不为 0 转,
        inc r28         ;+1
        rjmp main       ;返回主程序
```

7.3.5 廉价的 A/D 转换器

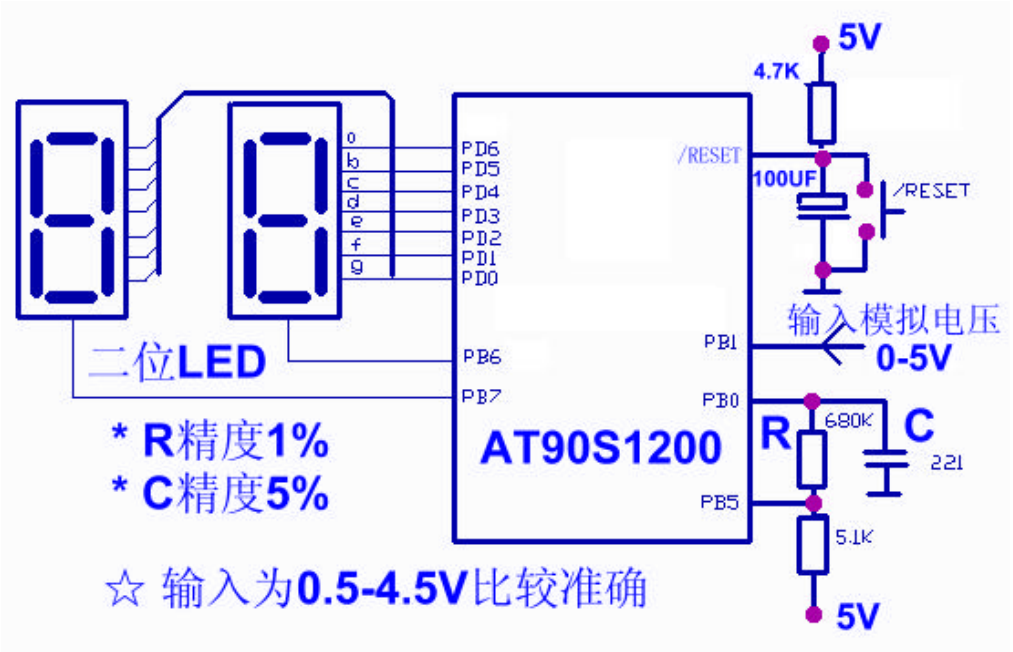
源程序:SLAVR735.ASM

★ 必须按下图接线才能正常工作!

AVR 单片机的 AT90 系列片内置有模拟比较器。这一节介绍用 AT90S1200 单片机实现廉价 A/D 转换器

一、硬件设计

使用 AVR 单片机及一个外部电阻和一个外部电容器设计成一个 A/D 转换器,并使用片内的定时器/计数器中断和模拟比较器中断。采用 RC 模拟转换原理。这程转方法在精确度和转换时间的花费上是较低的,适用一般要求不高的场合。硬件连接如图 6.1

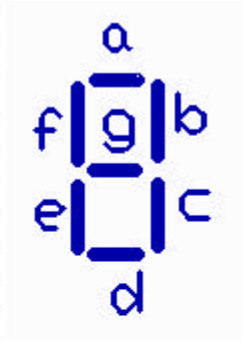


二、软件编程 (源程序为:模拟比较 AD.ASM)

```
;********************AVR 单片机实用实验程序 ************************
;* 标题:     廉价的 A/D 转换器
;* 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.Z
;* 程序适用于所有 AT90S 系列单片机
;*********************************************************************

;*********************************************************************
;* 硬件电路及说明阅<<廉价的 A/D 转换器>>一文
;* 本程序实测调试通过
;*********************************************************************
.include"1200def.inc"    ;应用器件配置文件,*.ASM 所在文件夹中不能缺,不然汇编出错提示
.org $0000
    rjmp reset       ;复位处理
.org $0002           ;EXT_INT0 外部中断入口地址
    rjmp inter           ;转 T0 溢出中断服务程序
.org $0003           ;TIMI_CAPT 定时器外部中断入口地址
    rjmp inter           ;转模拟比较器处理中断服务程序
.org $0010           ;主程序
reset:  ldi r20,$ff      ;设置 D 口为输出
    out ddrd,r20     ;送 D 口方向寄存器
```

```
sbi $18,7        ;置 I/O 寄存器 PORTB 的 7 位,LED 数目管十位片选
sbi $18,6        ;置 I/O 寄存器 PORTB 的 6 位,数目管个位片选
```

h	a	b	c	d	e	f	g	十六进制码	字形	R
1	1	1	1	1	1	1	0	FEH	0	R0
1	0	1	1	0	0	0	0	B0H	1	R1
1	1	1	0	1	1	0	1	EDH	2	R2
1	1	1	1	1	0	0	1	F9H	3	R3
1	0	1	1	1	0	1	1	B3H	4	R4
1	1	0	1	1	0	1	1	DBH	5	R5
1	1	0	1	1	1	1	1	DFH	6	R6
1	1	1	1	0	0	0	0	F0H	7	R7
1	1	1	1	1	1	1	1	FFH	8	R8
1	1	1	1	0	0	1	1	F3H	9	R9
1	1	1	1	0	1	1	1	F7H	A	R10
1	0	0	1	1	1	1	1	9FH	B	R11
1	1	0	0	1	1	1	0	CEH	C	R12
1	0	1	1	1	1	0	1	BDH	D	R13
1	1	0	0	1	1	1	1	CFH	E	R14
1	1	0	0	0	1	1	1	C7H	F	R15



硬件设定高电平

LED数目管点亮

```
ldi r20,$fe      ;R0-R15 存放显示字符.见表 6.1
mov r0,r20       ;0
ldi r20,$b0
mov r1,r20       ;1
ldi r20,$ed
mov r2,r20       ;2
ldi r20,$f9
mov r3,r20       ;3
ldi r20,$b3
mov r4,r20       ;4
ldi r20,$db
mov r5,r20       ;5
ldi r20,$df
mov r6,r20       ;67
ldi r20,$f0
mov r7,r20       ;7
ldi r20,$ff
mov r8,r20       ;8
ldi r20,$f3
mov r9,r20       ;9
ldi r20,$f7
mov r10,r20      ;A
ldi r20,$9f
mov r11,r20      ;B
ldi r20,$ce
mov r12,r20      ;C
ldi r20,$bd
```

```
    mov r13,r20      ;D
    ldi r20,$cf
    mov r14,r20      ;E
    ldi r20,$c7
    mov r15,r20      ;F

main:   rcall conini ;初始化 A/D 转换器
    sei              ;开中断
    ldi r16,$fc      ;置 B 口为输出,PB.0,PB.1 为输入,0B1111 1100
    out ddrb,r16
ddelay: clr r16          ;延时,清暂存计数器 1
        ldi r17,$f1  ;复位暂存计数器 2
 loop:  inc r16      ;清暂存计数器 1,加 1 计数
    brne loop            ;检查暂存计数器 1 不为 0 转,为 0 顺执
    inc r17          ;清暂存计数器 2,加 1 计数
    brne loop            ;检查暂存计数器 2 不为 0 转,为 0 顺执
    rcall adconv     ;调用启动转换器
wait:   brtc wait        ;等待中断,T 标志被清零转移,为 1 顺执
    clt                  ;清 T 标志
        rcall fetch      ;调用取显示
        ldi r20,$38      ;反复显示次数
start:  mov r30,r28          ;R28 送 Z 寄存器低位
display:andi r30,$f0     ;显示高位,与,即屏蔽高位,
    swap r30             ;交换半字节, 取得高位数据地址
ledh:   ld r25,z         ;取 LED 高位数据
    out portd,r25        ;高位显示送 D 口
    sbi $18,6            ;置位,关低位 LED 显示,
    cbi $18,7            ;清零,硬件设定低电平选中高位 LED
    ldi r27,$10      ;延时常数$10
delay1: dec r26          ;延时
    brne delay1
    dec r27
    brne delay1
    sbi $18,7            ;置位,关高位 LED
    nop
    mov r30,r28      ;显示低位
    andi r30,$0f
ledl:   ld r25,z
    out portd,r25
    sbi $18,7            ; 置位,关高位 LED 显示,
    cbi $18,6            ; 清零,硬件设定低电平 LED 亮,
    ldi r27,$10      ;延时常数$10
delay2: dec r26        ;延时
    brne delay2
```

```
    dec r27
    brne delay2
    sbi $18,6           ;关低位 LED
    nop
        dec r20     ;-1,
    brne start      ;不为 0 转,原(R20)=$38 即扫描显示 38 次
    rjmp main       ;返回主程序
inter:  in r28,tcnt0    ;中断服务程序
    clr r16         ;关闭 T0
    cli
    out tccr0,r16
    cbi portb,5
    set

    reti
conini: ldi r16,$0b         ;
    out acsr,r16
        ldi r16,$02
    out timsk,r16
        cbi portb,5
    ret

adconv: ldi r16,$40         ;
    out tcnt0,r16
    clt
        ldi r16,$02
    out tccr0,r16
    sbi portb,5
    ret

fetch:  mov r18,r28     ;取显示
        swap r18        ;半字节交换
        andi r18,$0f    ;与,屏蔽低位
        dec r18         ;-1
        dec r18
        dec r18
        dec r18
        brne l50        ;R18 不为 0 转
        mov r18,r28     ;
        andi r18,$0f    ;与,屏蔽低位
        rjmp fee
   l50: dec r18         ;
        brne l60
        mov r18,r28
```

```
        andi r18,$0f
        ori r18,$10
        rjmp fee
   l60: dec r18          ;
        brne l70
        mov r18,r28
        andi r18,$0f
        ori r18,$20
        rjmp fee
   l70: dec r18      ;-1
        brne l80     ;不为 0 转,为 0 顺执
        mov r18,r28
        andi r18,$0f;与
        ori r18,$30 ;或
        rjmp fee
   l80: ldi r28,$ff      ;(R28)=$FF
        ret

   fee: cbi eecr,0       ;清 I/O 寄存器 EEPROM 控制寄存器 EECR 的 0 位,设为读操作
        out eear,r18;R18 送 EEPROM 地址口,输出地址
        sbi eecr,0       ;置位 I/O 寄存器 EEPROM 控制寄存器的 0 位,读选通
        in r28,eedr ;获得数据,EEPROM 数据寄存器内容送 R28
        ret

.eseg
.org $0000           ; EEPROM 数据地址首址
.db 0x00,0x00,0x03,0x14,0x21,0x25,0x2f,0x33
.db 0x38,0x3f,0x47,0x52,0x59,0x6a,0x78,0x7d
.db 0x81,0x86,0x8b,0x90,0x9a,0x9f,0xa4,0xa7
.db 0xac,0xaf,0xbe,0xc5,0xc9,0xca,0xce,0xd0
.db 0xd3,0xd5,0xd7,0xd9,0xda,0xdf,0xe0,0xe1
.db 0xe2,0xe3,0xe4,0xe5,0xe6,0xe8,0xe9,0xec
.db 0xed,0xee,0xef,0xf0,0xf3,0xf4,0xf5,0xf6
.db 0xf8,0xf9,0xfa,0xfc,0xff,0xff,0xff,0xff
```

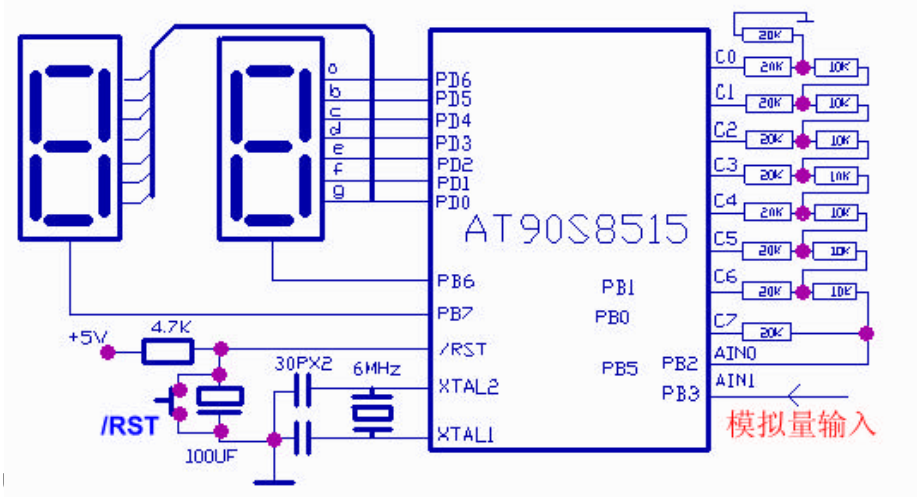
7.3.6 高精度廉价的 A/D 转换器

--- 用网络电阻组成的高精度 A/D 转换器

源程序:SLAVR736.ASM

★ 必须按下图接线才能正常工作!

```
;*****************************AVR 单片机实用程序 ************************
;* 标题:     高精度廉位的 A/D 转换器
;* 版本:     1.0
;* 最后更新日期:2000.08.08
;* 支援 E-mail: gzsl@sl.com.cn
;* 描述
;*  用网络电阻实现高精度廉位的 A/D 转换,本程序实测调试通过
;* 作者: SL.Z
;* 程序适用于所有单片机
```




```
.incl
.org $0000
    rjmp reset
.def temp=r16
.def temp1=r17
.equ label=$0100     ;字型表首址
.org $0010
reset:  ldi r20,$02      ;设堆栈指针
    out sph,r20
    out spl,r20
    ldi r20,$ff      ;设置 D 口、C 口为输出
    out ddrd,r20
    out ddrc,r20
    ldi r20,$f0          ;设 PB7~PB4 为输出,PB3~PB0 为输入
    out ddrb,r20
    out portb,r20
    clr r20              ;清 C 口
    out portc,r20
```

```
    sbi $18,7       ;关显示,硬件设定片选 LED 数目管低电平亮,高电平关
    sbi $18,6
    cli                 ;关中断
    ldi zh,high(label*2);
main:   ldi temp,$00
    nop
loop1:  out portc,temp
    nop
    nop
    nop
    in temp1,acsr       ;读模拟比较器控制和状态寄存器
    sbrs temp1,5
    rjmp naco       ;模拟比较器输出为 0 转
    rjmp haco       ;模拟比较器输出为 1 转
naco:   inc temp;+1
    brne loop1      ;不为 0 转
    ldi temp,$ff
haco:   mov r28,temp;暂存
    ldi r20,$38
display:mov temp,r28     ;显示十位字型
    andi temp,$f0
    swap temp
    clr zl
    add zl,temp
ledh:   lpm             ;取十位字型
    out portd,r0
    sbi $18,6           ;关 PB.6 硬件设定片选 LED 数目管高电平关，低电平开(灯亮)
    cbi $18,7           ;开 PB.7
    rcall delay
    mov temp,r28     ;显示个位
    andi temp,$0f
    clr zl
ledl:   add zl,temp
    lpm             ;取个位字型
    out portd,r0
    sbi $18,7       ;关 PB.7 硬件设定片选 LED 数目管高电平关，低电平开(灯亮)
    cbi $18,6       ;开 PB.6
    rcall delay     ;调用延时
        dec r20
    brne display
    rjmp main
delay:  ldi r27,$10 ;延时子程序
delay1: dec r26
    brne delay1
```

```
    dec r27
    brne delay1
    sbi $18,7          ;关 PB.7
    ret                ;子程序返回
```

h	a	b	c	d	e	f	g	十六进制码	字形
1	1	1	1	1	1	1	0	FEH	0
1	0	1	1	0	0	0	0	B0H	1
1	1	1	0	1	1	0	1	EDH	2
1	1	1	1	1	0	0	1	F9H	3
1	0	1	1	1	0	1	1	B3H	4
1	1	0	1	1	0	1	1	DBH	5
1	1	0	1	1	1	1	1	DFH	6
1	1	1	1	0	0	0	0	F0H	7
1	1	1	1	1	1	1	1	FFH	8
1	1	1	1	0	0	1	1	F3H	9
1	1	1	1	0	1	1	1	F7H	A
1	0	0	1	1	1	1	1	9FH	B
1	1	0	0	1	1	1	0	CEH	C
1	0	1	1	1	1	0	1	BDH	D
1	1	0	0	1	1	1	1	CFH	E
1	1	0	0	0	1	1	1	C7H	F

硬件设定高电平

LED数目管点亮

```
.cseg
.org $0100          ;字型表首址
.dw 0xb0fe,0xf9ed,0xdbb3,0xf0df
.dw 0xf3ff,0x9ff7,0xbdce,0xc7cf
```

7.3.7 星星灯

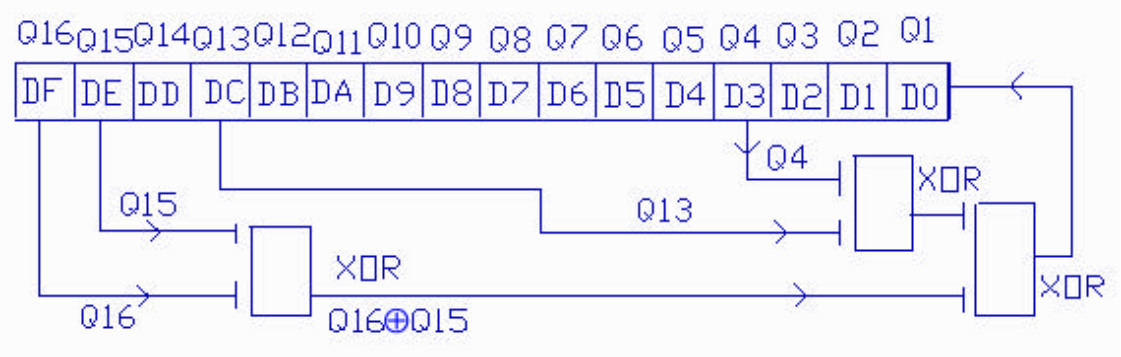
源程序:SLAVR737.ASM

用 AVR 单片机 8 位数据产生随机数,由 PORTA 口及 PORTC 口输出随机数,在 8X8 LED 上显示,硬件接线电路见“7.3.8 按钮猜数”。随机数的种子由程序设定(也可外接开关设定),启动种子后,由移位寄存器以互斥的异或逻辑组合返回循环产生。

```
.include"8515def.inc"
    rjmpRESET
.def    temp    =r16    ;暂存器
.deftemp1   =r17    ;暂存器 1
.defudata   =r21    ;存随机数送 A 口
.defddata   =r22    ;存随机数送 C 口
.cseg
.org    0x10
RESET:  ldi temp,high(RAMEND);设堆栈指针
    out SPH,temp
    ldi temp,low(RAMEND)
    out SPL,temp
```

```
        ldi     temp,0xff       ;设 A 口、C 口为输出
        out     ddra,temp       ;送方向寄存器 A
        out     ddrc,temp       ;送方向寄存器 C
start:  wdr                     ;关看门狗
        ldi     udata,0x6a      ; 设置随机数初值
        ldi     ddata,0x3c      ;
startp: out     porta,udata     ;输出到 A 口
        out     portc,ddata     ;输出到 C 口
        ldi     temp,0x80       ;设延时常数
        rcall   delay           ;调用延时子程序
        rcall   randm           ;调用十六位随机数子程序
        rjmp    startp
delay:                  ; 通用延时子程序从略
....
```

16 位移位产生随机数原理图



8--16 位移位寄存器产生随机数循环组合

位 数	循环输入组合 S=2^n-1 Qn XOR Qm
8	Q2 ⊕ Q3 ⊕ Q4 ⊕ Q8 (现程序按钮猜数采用 8 位数)
9	Q5 ⊕ Q9
10	Q7 ⊕ Q10
11	Q9 ⊕ Q11
12	Q2 ⊕ Q10 ⊕ Q11 ⊕ Q12
13	Q1 ⊕ Q11 ⊕ Q12 ⊕ Q13
14	Q2 ⊕ Q12 ⊕ Q13 ⊕ Q14
15	Q14 ⊕ Q15
16	Q4 ⊕ Q13 ⊕ Q15 ⊕ Q16

```
randm:                          ;产生十六位随机数子程序
          mov   temp,udata      ;产生 A 口随机数
          mov   temp1,udata     ;
          rol   temp            ;通过进位位左循环移位
          eor   temp1,temp      ;异或
```

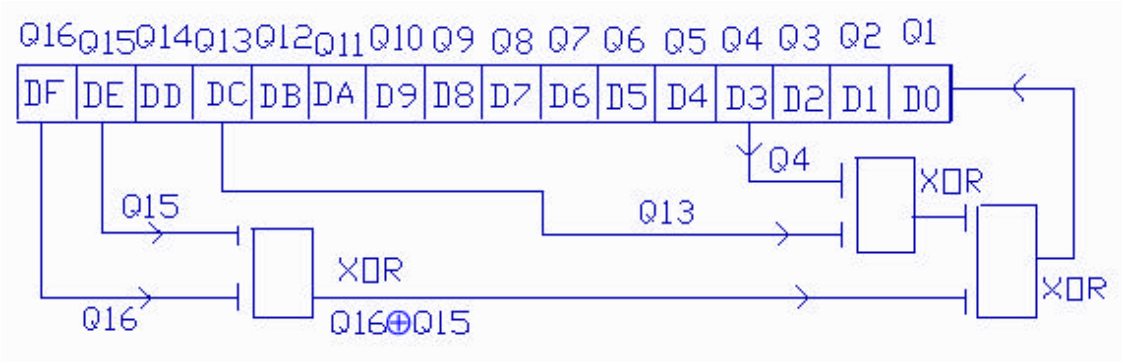
```
rol   temp        ; 通过进位位左循环移位
rol   temp        ; 通过进位位左循环移位
eor   temp1,temp  ;异或
mov   temp,ddata  ; 产生 C 口随机数
swap  temp        ; 通过进位位左循环移位
eor   temp,temp1  ;异或通过进位位左循环移位
rol   temp        ; 通过进位位左循环移位
rol   ddata       ; 通过进位位左循环移位
rol   udata       ; 通过进位位左循环移位
ret               ;子程序返回
```

7.3.8　按钮猜数程序

源程序:SLAVR738.ASM

许多场合如按钮猜数(电脑摇奖,电脑选出幸运号),游戏开始按钮等待一个不规则且不定序的数据产生,即须要随机数发生器。随机数的种子由程序设定(也可外接开关设定),启动种子后,由移位寄存器以互斥的异或逻辑组合返回循环产生。产生随机数的原理图如下:

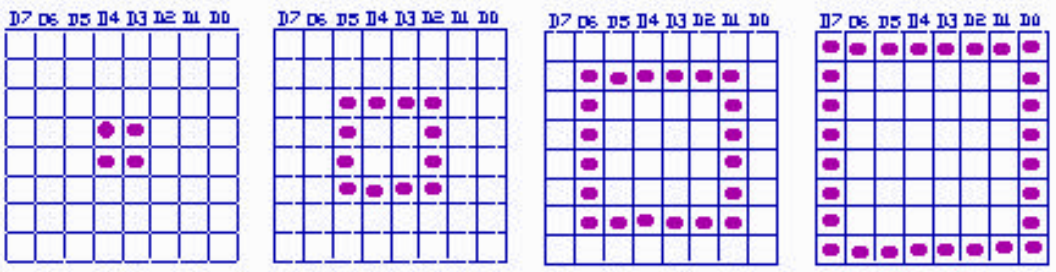
16 位移位产生随机数原理图

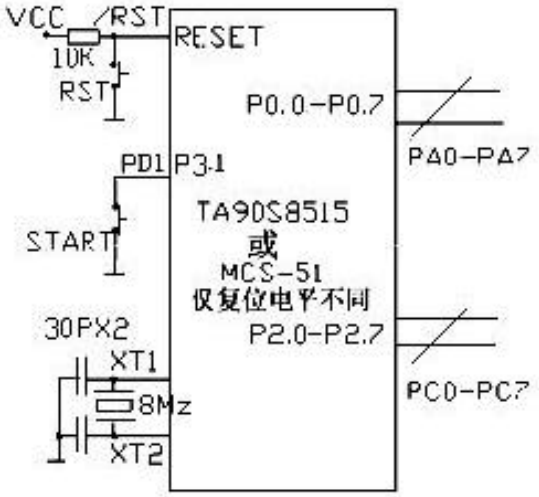


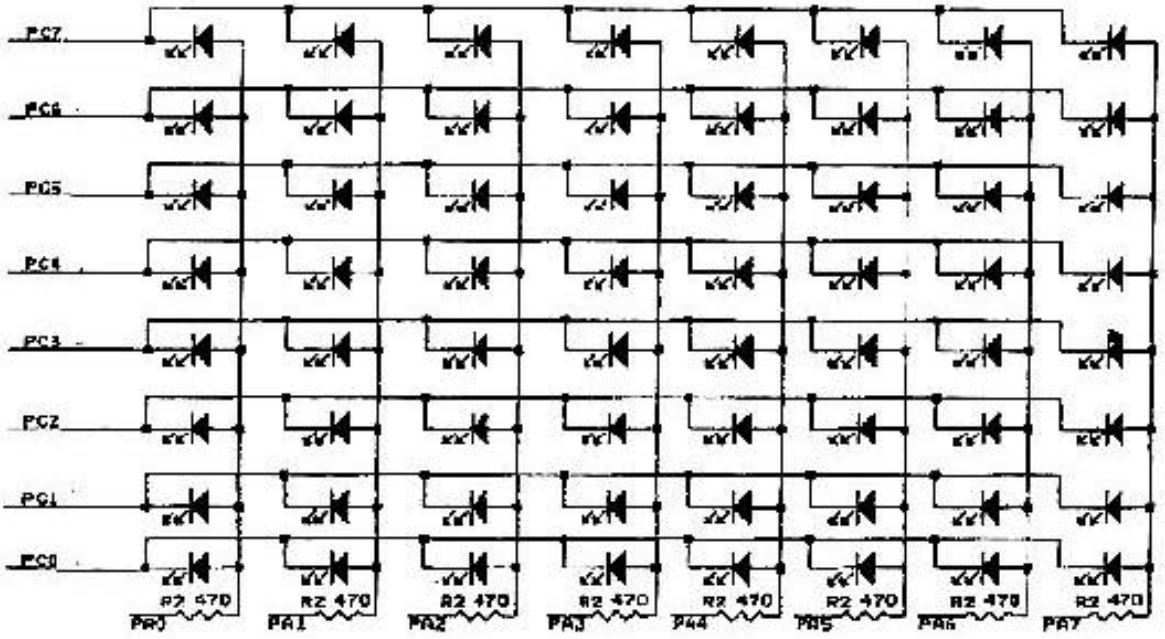
8—16 位移位寄存器产生随机数循环组合

位 数	循环输入组合 S=2^n-1 Qn XOR Qm
8	Q2 ⊕ Q3 ⊕ Q4 ⊕ Q8 (现程序按钮猜数采用 8 位数)
9	Q5 ⊕ Q9
10	Q7 ⊕ Q10
11	Q9 ⊕ Q11
12	Q2 ⊕ Q10 ⊕ Q11 ⊕ Q12
13	Q1 ⊕ Q11 ⊕ Q12 ⊕ Q13
14	Q2 ⊕ Q12 ⊕ Q13 ⊕ Q14
15	Q14 ⊕ Q15
16	Q4 ⊕ Q13 ⊕ Q15 ⊕ Q16

以 8X8 LED 阵列,开机时为了避免被使用者预测出压按时间对应随机数的变化值,故 LED 字幕以广告动画画面显示,并令随机数随着变化使无法预测随机数起始值,广告动画面共有四张,每张有 8 位数据。见”org　dpfstb”;







AVR 直接驱动 8X8 LED 按钮猜数电路及接线配置图

由按钮(PD1)按下,AVR 用 8 位数据产生随机数,由 PORTA 口及 PORTC 口输出随机数,在 8X8 LED 上显示好玩的真实的按钮猜数。

```
.include "8515def.inc"
.def    peed    =r16
.def    dspn    =r17    ;存显示初始动画次数
.def    temp2   =r18
```

```
.def    temp1   =r19
.def    temp    =r20
.def    scndp   =r21
.def    cnt     =r22
.def    rdata   =r23     ;存随机种子数
.def    rdata9  =r24
.equ    dpfstb  =0x01e0      ;大小矩形图表首址
.equ    randtb  =0x0210      ;随机数种子表首址
.equ    numbertb=0x0240      ;0-9 数字表首址
.org  $0000
    rjmpRESET        ;Reset Handle
.cseg
.org $0010
RESET:  ldi     peed,high(RAMEND)    ;设置堆栈$25F,见器件配置文件"8515def.inc"
        out     SPH,peed
        ldi     peed,low(RAMEND)
        out     SPL,peed
        ldi     peed,0xff     ;对口初始化,
        out     ddra,peed     ;设 A 口为输出
        out     ddrc,peed     ;设 C 口为输出
        ldi     peed,0xfd     ;PD1 作输入,且接内部上拉电阻
        out     ddrd,peed     ;PD1 为输入,其余为输出
        ldi     peed,0xff     ;关 D 口
        out     portd,peed
        ldi     peed,0x13     ;显示画面次数
start:  ldi     dspn,0x06     ;显示初始动画
        ldi     zh,high(dpfstb*2)
        ldi     zl,low(dpfstb*2)
dspfm:  rcall   ldtb8         ;调用程序区数送到内存 RAM
        ldi     temp2,0xa0    ;显示动画面次数
dspfm1: rcall   scan1         ;调用从内存取数显示一次
        sbis    pind,01       ;I/O 口的位被置位跳行,检测到 PD1 按下否
        rjmp    getseed       ; 检测到 PD1 按下转
        dec     temp2         ;-1
        brne    dspfm1        ;不为 0 转
        dec     dspn     ; 初始画面次数-1
        brne    dspfm         ; 不为 0 转
        rjmp    start         ;转到显示初始动画
getseed:inc     temp          ;+`1,根据 PD1 按下的时间,选择随机数种子
        sbis    pind,01       ; I/O 口的位被置位跳行,检测到 PD1 按下否
        rjmp    getseed       ; 检测到 PD1 按下,继续计数
        andi    temp,0x1f     ;按钮松开,取随机数种子与 0X0F 加
        ldi     zh,high(randtb*2)
        ldi     zl,low(randtb*2)
```

```
        add     zl,temp
        lpm
        mov     rdata,r0     ;得到随机数种子
next:   ldi     dspn,0x08    ;显示 8 个不同的随机数;
repeet: rcall   randm        ;调用产生随机数子程序
        rcall   dspnumber    ;调用显示 8 个不同的随机数
        dec     dspn     ;-1
        brne    repeet       ;dspn 不为 0 转
        rcall   randm        ; 调用产生随机数子程序
guess1: rcall   dspnumber    ;调用显示同一随机数,直到有键按下
        sbic    pind,01      ;松开后再往下执行(I/O 口清零跳行)
        rjmp    guess1       ;转显示同一随机数,直到有键按下
wait:   rcall   dspnumber    ;
        sbis    pind,01
        rjmp    wait         ;等待按钮按下
        ldi     rdata9,0x03      ;显示动画三次
start0: ldi     dspn,0x06        ;每次显示六幅画面
        ldi     zh,high(dpfstb*2)
        ldi     zl,low(dpfstb*2)
dspfm0: rcall   ldtb8            ;调用从 Z 指向的程序区取数据送到内存 0080-0087 中
        ldi     temp2,0xa0       ;显示次数
dspfm1a:rcall   scan1            ;调用从内存 0080-0087 中取数据显示一次
        dec     temp2            ;-1
        brne    dspfm1a          ;不为 0 转
        dec     dspn         ;显示初始动画次数-1
        brne    dspfm0           ;不为 0 转
        dec     rdata9           ;显示动画三次-1
        brne    start0           ;不为 0 转
        rjmp    next              ;转显示 8 个不同的随机数
dspnumber:                        ;显示一个 0-9 数字的子程序
        ldi     zh,high(numbertb*2)
        ldi     zl,low(numbertb*2)
        add     zl,rdata9
        rcall   ldtb8            ;取数
        ldi     temp2,0xa0    ;该数字重复显示 A0H 次
dspn1:  rcall   scan1
        dec     temp2
        brne    dspn1
        ret
scan1:  push    xl                ;从内存 0080-0087 中取数据显示一次
        ldi     temp,0b01111111
        mov     scndp,temp
        ldi     cnt,0x08
col1:   out     portc,scndp      ;显示屏幕的一列
```

```
        ld      r1,x+
        out     porta,r1
        rcall   delay
        sec
        ror     scndp
        dec     cnt
        brne    col1
        pop     xl
        ret
ldtb8:  ldi     xl,0x80      ;从 Z 指向的程序区取数据送到内存 0080-0087 中
        ldi     xh,0x00
        ldi     temp1,0x08
        push    xl
nexld1: lpm
        st      x+,r0
        ld      r0,z+
        dec     temp1
        brne    nexld1
        pop     xl
        ret
delay:           ;通用延时子程序从略
....
randm:  mov     temp,rdata     ;产生 8N(0≤N≤9)随机数子程序
        mov     temp1,rdata
        swap    temp1
        eor     temp,temp1
        rol     temp1
        eor     temp,temp1
        rol     temp1
        eor     temp,temp1
        rol     temp
        rol     rdata
        mov     rdata9,rdata
        andi    rdata9,0x0f
        cpi     rdata9,0x0a
        brsh    randm          ;产生了一个 0≤RDATA9≤9 的随机数
        lsl     rdata9
        lsl     rdata9
        lsl     rdata9
        ret

.cseg
.org      dpfstb;            ;大小方框字形表
;small  o
```

```
.db     0b00000000,0b00000000,0b00000000,0b00011000
.db     0b00011000,0b00000000,0b00000000,0b00000000
.db     0b00000000,0b00000000,0b00111100,0b00100100
.db     0b00100100,0b00111100,0b00000000,0b00000000
.db     0b00000000,0b01111110,0b01000010,0b01000010
.db     0b01000010,0b01000010,0b01111110,0b00000000
;big   o
.db     0b11111111,0b10000001,0b10000001,0b10000001
.db     0b10000001,0b10000001,0b10000001,0b11111111
.db     0b00000000,0b01111110,0b01000010,0b01000010
.db     0b01000010,0b01000010,0b01111110,0b00000000
.db     0b00000000,0b00000000,0b00111100,0b00100100
.db     0b00100100,0b00111100,0b00000000,0b00000000
.cseg
.org    randtb                  ;随机数种子表
.db     0x5a,0x7b,0x5b,0x4f,0x66,0x6d,0x7d,0x07
.db     0x3b,0x8c,0x67,0x9a,0x99,0x7e,0x2d,0x3e
.db     0x5c,0x6d,0x5b,0x7e,0xf6,0xe7,0x4c,0xc8
.db     0x69,0x9c,0xe2,0x75,0x6c,0xd3,0xe8,0x9a
.cseg
.org    numbertb                ;0-9 数字字形表
;0
.db     0b00111000,0b01000100,0b01000100,0b01000100
.db     0b01000100,0b01000100,0b01000100,0b00111000
;1
.db     0b00010000,0b00011000,0b00010000,0b00010000
.db     0b00010000,0b00010000,0b00010000,0b00111000
;2
.db     0b00011100,0b00100010,0b00100000,0b00010000
.db     0b00001000,0b00000100,0b00000010,0b00111110
;3
.db     0b00111100,0b00010000,0b00001000,0b00010000
.db     0b00100000,0b00100000,0b00100010,0b00011100
;4
.db     0b00100000,0b00110000,0b00101000,0b00100100
.db     0b00100010,0b11111110,0b00100000,0b00100000
;5
.db     0b01111110,0b00000010,0b00111110,0b01000000
.db     0b01000000,0b01000000,0b01000010,0b00111100
;6
.db     0b00110000,0b00001000,0b00000100,0b00111100
.db     0b01000100,0b01000100,0b01000100,0b00111000
;7
.db     0b01111100,0b01000000,0b00100000,0b00010000
```

```
.db     0b00001000,0b00001000,0b00001000,0b00001000
;8
.db     0b00111000,0b01000100,0b01000100,0b00111000
.db     0b01000100,0b01000100,0b01000100,0b00111000
;9
.db     0b00111000,0b01000100,0b01000100,0b01111000
.db     0b01000000,0b01000000,0b01000100,0b00111000
```

7.3.9 汉字的输入

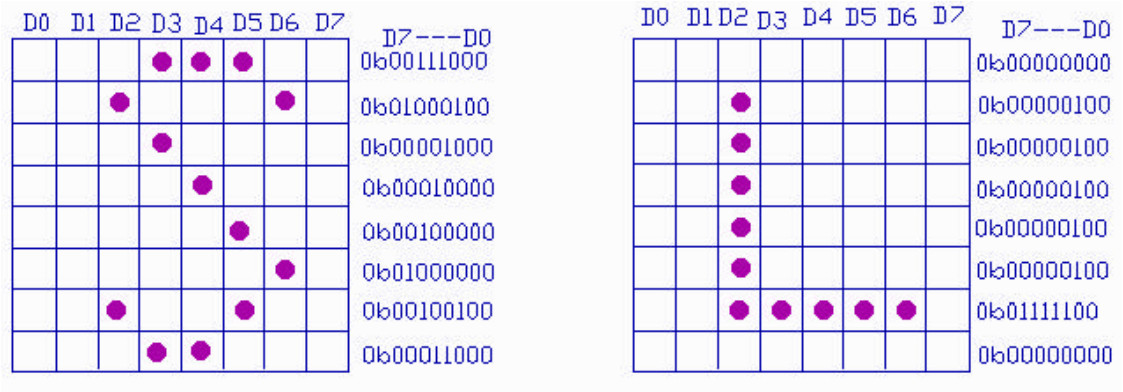
源程序:SLAVR739.ASM

硬件电路见“7.3.8 按钮猜数程序”

```
;**********************AVR 单片机实验测试程序 **********************
;* 标题:    汉字的输入
;* 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.Z
;* 程序适用于所有单片机
;*****************************************************************

.include "8515def.inc"
.def    dspn    =r23
.def    temp2   =r24
.def    temp1   =r17
.def    temp    =r18
.def    scndp   =r19
.def    cnt     =r20
.equ    dpfstb  =0x01e0
.org    $0000
   rjmp RESET          ;Reset Handle
.org $0010
RESET:
   ldi   r16,high(RAMEND)     ;设堆栈为$025F
   out   SPH,r16
   ldi   r16,low(RAMEND)
   out   SPL,r16
         ldi     r16,0xff     ;设 A 口、C 口为输出
         out     ddra,r16     ;A 口方向寄存器
         out     ddrc,r16     ;C 口方向寄存器
```

```
dspfst: ldi     dspn,0x07               ;显示次数
        ldi     zh,high(dpfstb*2)       ;高位取数
        ldi     zl,low(dpfstb*2)        ;低位取数
dspfm:  rcall   ldtb8                   ;调用取字形子程序
        ldi     temp2,0xa0;循环次数
dspfm1: rcall   scan1
        dec     temp2
        brne    dspfm1
        dec     dspn
        brne    dspfm
        rjmp    dspfst
scan1:  push    xl                      ;XL 进栈
        ldi     temp,0b01111111;  第一次选中 PC.7,硬件设定低电平 LED 亮
        mov     scndp,temp
        ldi     cnt,0x08                ;取一个字符需 8 次取数
col1:   out     portc,scndp  ;选通数据送 C 口,第一次选中 PC.7
        ld      r1,x+        ;取数后地址指针加 1
        out     porta,r1     ;数据送 A 口
        ldi     r16,0x10     ;送延时常数
        rcall   delay                   ;调用延时
        sec                             ;置位进位位
        ror     scndp        ;通过进位位右循环
        dec     cnt          ;-1
        brne    col1                    ;不为 0 转
        pop     xl           ;为 0,XL 出栈
        ret                             ;子程序返回
ldtb8:  ldi     xl,0x80      ;取数(字形)子程序
        ldi     xh,0x00      ;
        ldi     temp1,0x08   ;取数(字符)次数
        push    xl                      ;XL 进栈
nexld1: lpm                  ;从程序存储器取数,将 Z 寄存器指向的一个字节装入 R0
        st      x+,r0                   ;X 寄存器内容(字形)送 R0,后 X 指针加 1
        ld      r0,z+                   ; Z 寄存器内容(字形)送 R0,后 Z 指针加 1
        dec     temp1        ;-1
        brne    nexld1       ;未完继续取数
        pop     xl           ;XL 出栈
        ret                  ;子程序返回
delay:                       ;通用延时子程序从略
.org    dpfstb       ;
```

;在 8X8 的方格中填字,硬件设定高电平为 1 点亮 LED,注意:编码的高、低位与习惯编码书写方法正好相反,其目的为汉字、字符书写符合人们习惯(正写)!




```
;字符 S
.db     0b00111000,0b01000100,0b00001000,0b00010000
.db     0b00100000,0b01000000,0b00100100,0b00011000
;字符 L
.db     0b00000000,0b00000100,0b00000100,0b00000100
.db     0b00000100,0b00000100,0b01111100,0b00000000
;字符“双龙电子”字形表略
```

7.4.1　10 位 AD/转换

源程序:SLAVR741.ASM

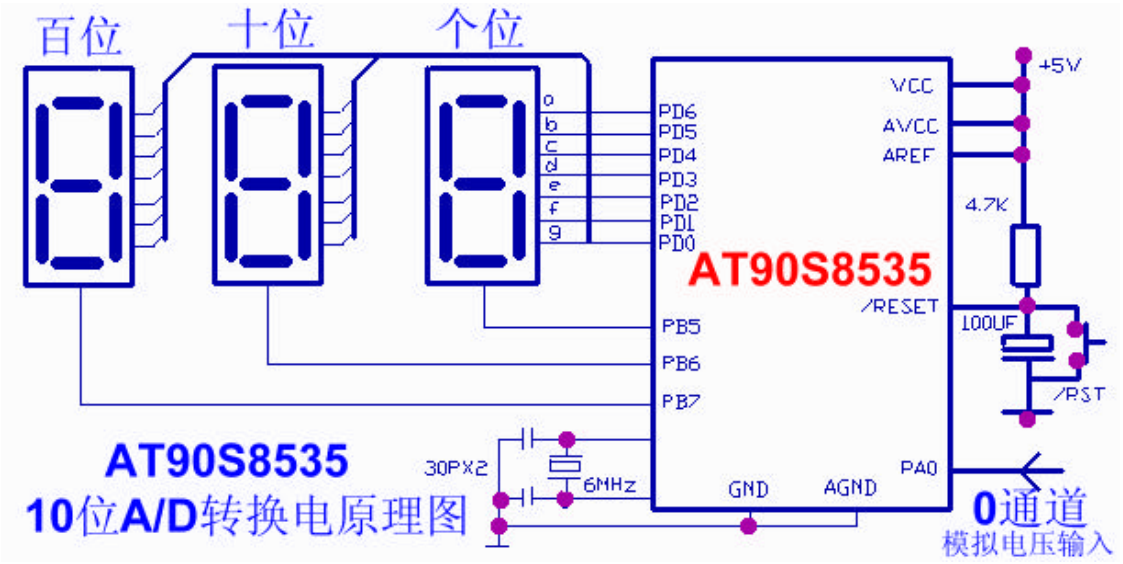
AVR 单片机中 AT90S8535 是功能较强的单片机.有如下特点:

1. AVR RISC 结构
2. AVR—高性能、低功耗 RISC 结构
   —118 条指令——大多数为单指令周期
   —32 个 8 位通用（工作）寄存器
   —工作在 8MHz 时具有 8MIPS 的性能
3. 数据和非易失性程序内存
   —4K/8K 字节的在线可编程 FLASH（擦除次数：1000 次）
   —256/512 字节 SRAM
   —256/512 字节在线可编程 EEPROM（寿命：100000 次）
   —程序加密位
4. 外围（Peripheral）特点
   —两个具有比较模式的可预分频（Prescale）8 位定时器/计数器
   —一个可预分频、具有比较、捕捉和两个 8/9/10 位 PWM 功能的 16 位定时器/计数器
   —片内模拟比较器
   —可编程的看门狗定时器（由片内振荡器生成）
   —8 通道 10 位 ADC
   —全双工 UART
5. 特别的 MCU 特点
   —上电复位电路
   —具有记数功能、有独立振荡器的实时时钟（RTC）
   —低功耗空闲、省电和掉电模式
   —内外部中断源

6. 4MHz、3V、20℃条件下的功耗:
   —工作模式: 6.4mA
   —空闲模式: 1.9mA
   —掉电模式: <1μA
7. I/O 和封装
   —32 个可编程的 I/O 脚
   —40 脚 PDIP、PLCC 和 TQFP 封装
8. 工作电压
   —2.7V-6.0V（AT90LS4434 和 AT90LS8535）
   —4.0V-6.0V（AT90S4434 和 AT90S8535）
9. 速度
   —0-4MHz（AT90LS4434 和 AT90LS8535）
   —0-8MHz（AT90S4434 和 AT90S8535）

```
;****************AVR 单片机实验测试程序 ************************
;* 标题:AT90S8535 的 10 位 A/D 转换器
;* 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.Z
;*  硬件电路及本程序实测调试通过
;*********************************************************************
```

★ 硬件电路必须按下图连接!

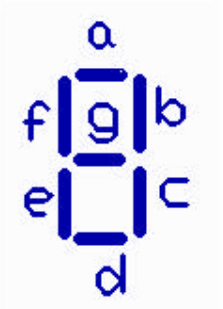



```
.include"8535def.inc"     ;AT90S8535 器件配置文件
.org $0000
    rjmp reset
.org $000e                ;INTER 中断入口地址
```

```
    rjmp inter
.defhledbyte=r19        ;存放 ADCH
.def    lledbyte=r18    ;存放 ADCL
.equ    label=$0100     ;字形表首址
.equ    distime=$38     ;显示次数
.def    temp=r16
reset:  ldi temp,$02    ;设堆栈指针$025F
    out sph,temp
    ldi temp,$5f
    out spl,temp
    ldi temp,$ff        ;B 口、D 口为输出
    out ddrb,temp
    out ddrd,temp
    out portd,temp      ;开通 LED 数目管,硬件设定高电平亮
    clr temp
    out ddra,temp       ;A 口为输入
    out porta,temp
main:   clt             ;清 T 标志
    sei                 ;开中断
    rcall conini        ;调用 A/D 初始化
wait:   brtc wait       ;等待中断
    clt
    ldi r20,distime
start:  ldi zh,high(label*2)
    mov temp,r19
    rcall outpd         ;取出显字形型
    cbi $18,7           ;显示百位 LED
    sbi $18,6           ; 关闭十位 LED
    sbi $18,5           ; 关闭个位 LED
    rcall delay         ;延时
    mov temp,r18
    swap temp
    rcall outpd
    cbi $18,6           ; 显示十位 LED
    sbi $18,7           ; 关闭百位 LED
    sbi $18,5           ; 关闭个位 LED
    rcall delay         ; 延时
    mov temp,r18
    rcall outpd
    cbi $18,5           ; 显示个位 LED
    sbi $18,6           ; 关闭十位 LED
    sbi $18,7           ; 关闭百位 LED
    rcall delay         ; 延时
    dec r20             ;显示次数减 1
```

```
    brne start          ;显示次数未完转,为 0 顺执
    rjmp main           ;返回主程序
inter:  in lledbyte,adcl;中断后,读取 ADC 数据寄存器低位数据
    in hledbyte,adch    ;取 ADC 数据寄存器高位数据
    cbi adsc,6          ;停止 ADC 转换
    set                 ;置 T 标志
    reti                ;中断返回
conini: ldi temp,$8d    ;设置 ADC 转换,中断触发,ADC 为单一模式且 32MCU 除频
    out adcsr,temp
    clr temp            ;
    out admux,temp      ;选择 0 通道
    sbi adcsr,6
    ret
delay:  ldi r27,$10     ;延时子程序
delay1: dec r26
    brne delay1
    dec r27
    brne delay1
    ret
outpd:  andi temp,$0f
    ldi zl,low(label*2)
    add zl,temp
    lpm                 ;取字符
    out portd,r0        ;输出字第
    ret
.cseg
.org $0100          ;字型表首地址
```

h	a	b	c	d	e	f	g	十六进制码	字形
1	1	1	1	1	1	1	0	FEH	0
1	0	1	1	0	0	0	0	B0H	1
1	1	1	0	1	1	0	1	EDH	2
1	1	1	1	1	0	0	1	F9H	3
1	0	1	1	1	0	1	1	B3H	4
1	1	0	1	1	0	1	1	DBH	5
1	1	0	1	1	1	1	1	DFH	6
1	1	1	1	0	0	0	0	F0H	7
1	1	1	1	1	1	1	1	FFH	8
1	1	1	1	0	0	1	1	F3H	9
1	1	1	1	0	1	1	1	F7H	A
1	0	0	1	1	1	1	1	9FH	B
1	1	0	0	1	1	1	0	CEH	C
1	0	1	1	1	1	0	1	BDH	D
1	1	0	0	1	1	1	1	CFH	E
1	1	0	0	0	1	1	1	C7H	F



硬件设定高电平

LED数目管点亮

```
.dw 0xb0fe,0xf9ed,0xdbb3,0xf0df
.dw 0xf3ff,0x9ff7,0xbdce,0xc7cf
```

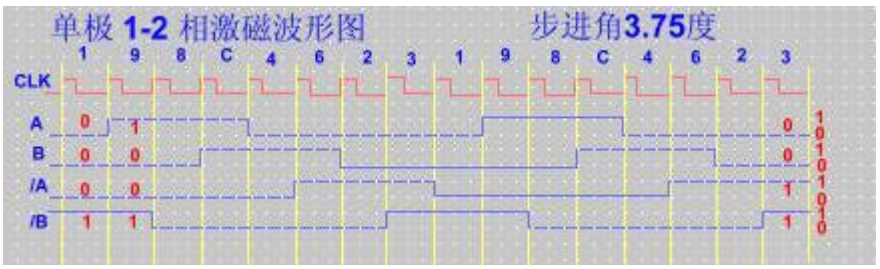
7.4.2 步进电机控制程序

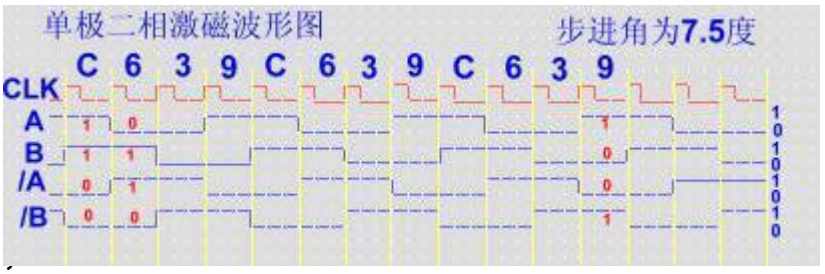
源程序:SLAVR742.ASM

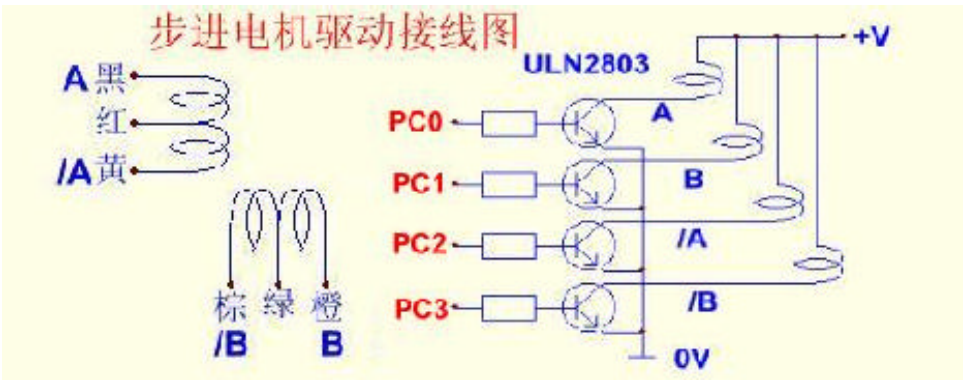
自从六十年代初期步进电机面世以来,在过去几年它的重要性大大提高了。它用来驱动时钟和其他采用指针的仪器,打印机、绘图仪、磁盘光盘驱动器、各种自动控制阀、各种工具,还有机器人等的机械装置。关于马进电机工作原理请参考有关资料。

下面用单极 1-2 相激磁方法步进电机做实验,即 1 极、2 极、1 极、2 极、....极以次循环,如何用单极二相激该方法控制步进电机,由读者或用户自行编制程序实验。

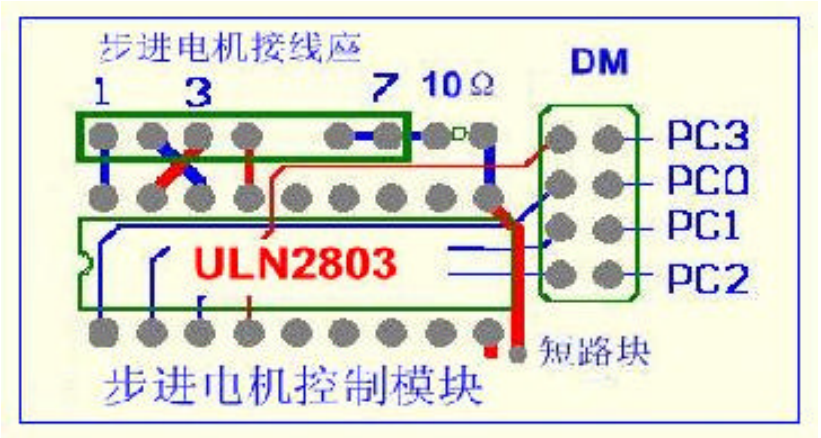
```
;实验选用 4.5V 步进电机,用 5V 即可,实验时节省一组步进电机驱动电源;
;型号:MA82135; 相数:2 相; 电压:4.5V; 电流/相:0.12A; 电阻欧姆:34Ω/相; 重量:30g
;
;
;
;
;
;
;
;
;
;
;
;
;
```







;



;
;
;
;
;
;
;
;
;
;
;

```
;*****************************************
;*    步进电机控制程序(单极 1-2 相)          *
;*                                         *
;*SLAVR742.ASM                             *
;*use ULN2803 ;使用 PC0-PC3 驱动步进电机     *
;*use 11-17new bord                        *
;*****************************************
.include"8515def.inc"
.def   temp    =r16
.def   dt      =r19
.def   np      =r17
.def   step    =r18
.def   TStep   =r20
.def   cnt     =r21
.equ   turntab=0x0200
.org  $0000
       rjmp   RESET
.cseg
.org  0x010
RESET:
       ldi  temp,low(RAMEND);设堆栈
       out  SPL,temp
       ldi  temp,high(RAMEND)
       out  SPL+1,temp
       ser   TEMP        ;C 口设置为输出
       OUT   ddrc,TEMP
       ldi   zl,low(turntab*2)   ;步进电机旋转资料指针
       ldi   zh,high(turntab*2)
       ldi   np,4
       ldi   temp,$44
       out   portc,temp      ;初始化
       ldi   TStep,$25
       rcall delay
       ldi   cnt,10
       clt
rep:   ldi   step,192
       ldi   TStep,1     ;1--255
       rcall turn
       dec   cnt
       brne  rep
loop:  nop
       rjmp  loop
;*************************************************************
;  t=1  uncircle turn   ;T=1 逆时针转                        *
```

```
;  t=0   circle  turn   ;T=0 顺时针转                          *
;   96  step  a  turn                                        *
;  TStep is  time  of a step  ;                              *
;************************************************************
turn:  brts uncircle;判转向
       inc   np      ;正转
       cpi   np,8
       brne  next
       clr   np
next:  push  zl
       add   zl,np
       lpm
       out   portc,r0
       pop   zl
       rcall delay
       dec   step
       brne  turn
       ret
uncircle:       ;反转
       dec   np
       cpi   np,$ff
       brne  next
       ldi   np,$07
       rjmp  next
delay:  push   TStep;延时子程序
del1:   ldi    dt,70
del2:   push   dt
del3:   dec    dt
        brne   del3
        pop    dt
        dec    dt
        brne   del2
        dec    TStep
        brne   del1
        pop    TStep
        ret
.org   turntab
;       0    1    2    3    4    5    6    7 ;步进电机旋转资料表
.db    0x11,0x99,0x88,0xcc,0x44,0x66,0x22,0x33
```

7.4.3 测脉冲宽度

源程序:SLAVR743.ASM

```
;本程序略加修改可应用于测速,测距,测频率等,用 LED 数目管显示
;本程序在 SL-AVR 开发下载实验器上验证通过
;硬件连接:AT90S8515 的 ICP 作输入,测量输入脉冲宽度时间,以μS/秒计算,
;由六位 LED 十进制显示
.include"8515def.inc"
    rjmp reset
.def   cnt1d =r03                ;cnt1d-cnt5d:存放十进制数
.def   cnt2d =r04                ; 存放形式(压缩 BCD 码)
.def   cnt3d =r05                ;从 cnt1 到 cnt5 依数存放个位、十位.....
.def   cnt4d =r06
.def   cnt5d =r07
.def   tmp1h =r08
.def   tmp2h =r09                ;tmp1h,tmp2h:存放第一次捕捉的 timer1 的值
.def   tim1l =r10
.def   tim1h =r11                ;存放 timer1 溢出的次数
.def   tmp5h =r12
.def   tmp6h =r13                ;tmp5h-tmp8h:存放二次捕捉 timer1 的差值
.def   tmp7h =r14
.def   tmp8h =r15
.def   temp  =r16                ;暂存器
.def   cnt4  =r17
.def   cntn  =r18
.def   cntn1 =r19
.def   tempn =r20                ;利用 tempn 的 d0 位判断是那一次捕捉
.def   tempn1=r21
.cseg
.org 0x03                        ;icp 触发中断向量
icpt1:
    rjmp  captr
.org 0x06                        ;timer1 溢出中断向量
intt1:
    rjmp  calts
.cseg
.org 0x10
captr:                           ;icp 触发中断子程序
    wdr
    sbrc  tempn,  00          ;判断是那一次捕捉
    rjmp  cap2
cap1:                            ;第一次捕捉处理
    in    tmp1h, icr1l
    in    tmp2h, icr1h        ;把捕捉的 timer1 的值放在 tmp1h 和 tmp2h
```

```
    ori   tempn, $01
    ldi   r16,   0b10001000
    out   timsk, r16          ;致 timer1 中断和捕捉中断
    reti
cap2:                         ;第二次捕捉处理
    in    tmp5h, icr1l
    in    tmp6h, icr1h        ;把捕捉的 timer1 的值放在 tmp5h 和 tmp6h
    rcall transd              ;
    rcall htd4                ;把 tmp5h-tmp8h 转成十进制
    ldi  tempn, $00          ;令 tempn d0=0
    rcall clrtmm
    ldi   r16,   0b00000000
    out   timsk, r16          ;除 timer1 中断和致捕捉中断
    ret
calts:                        ;timer1 溢出中断子程序
    inc   tim1l
    breq  ov
    rjmp  b
ov:     inc   tim1h
    brne  b
    set
b:  reti
transd:                          ;二次捕捉 timer1 的差值处理
    clc
    sbc   tmp5h, tmp1h           ;结果放在 tmp5h-tmp8h
    sbc   tmp6h, tmp2h
    brcc  posv
    dec   tim1h
    dec   tim1l
posv:
    mov   tmp7h, tim1l
    mov   tmp8h, tim1h
    ret
reset:
    ldi   temp,  low(ramend)
    out   spl,   temp
    ldi   temp,  high(ramend)      ;设置堆栈
    out   spl+1, temp
    ldi   temp,  $0e                ;设定 wdtcr 中 wde=1
    out   wdtcr, temp
    cli
    ldi   temp,  $ff               ;初始化数码管状态
    out   ddrb,  temp              ;B 口:数码管数据输出
    out   ddrd,  temp              ;D 口:pd0-pd5 为数码管片选
```

```
    out   portd, temp              ;数码管低电平选中
    ldi   temp,  $00
    out   portb, temp              ;共阴极,数码管全灭
    ldi   tempn1,00
    rcall clrtm
    ldi   temp,  $00
    out   tccr1a,temp
    out   tccr1b,temp
    out   tcnt1h,temp              ;装入计数值
    out   tcnt1l,temp              ;tcnt1h=tcht1l=00
    clt
    sei                            ;开中断
    ldi   r16,   0b00001000
    out   timsk, r16
    ldi   r16,   0b11000010
    out   tccr1b,r16               ;开始计数
reptw:
    sbrs  tempn1,00
    rjmp  reptw
    rjmp  reset
clrtm:
    clr   tmp1h
    clr   tmp2h
    clr   tmp5h
    clr   tmp6h
    clr   tmp7h
    clr   tmp8h
clrtmm:
    clr   tim1h
    clr   tim1l
    clr   cnt4
    clt
    ret
htd4:                              ;把 tmp5h-tmp8h 转成十进制子程序
    ldi   cntn,  32
    clr   cnt1d
    clr   cnt2d
    clr   cnt3d
    clr   cnt4d
    clr   cnt5d
    clc
loopd:
    rol   tmp5h
    rol   tmp6h
```

```
    rol   tmp7h
    rol   tmp8h
    rol   cnt1d
    rol   cnt2d
    rol   cnt3d
    rol   cnt4d
    rol   cnt5d
    dec   cntn
    breq  end
    rcall adjn
    rjmp  loopd
end:                        ;在数码管显出十进制数
    ldi   cntn,  $ff
end1:
    ldi   cntn1, $ff
end2:   mov   temp,  cnt1d
    andi  temp,  $0f        ;显示个位
    rcall a
    cbi   portd, 00
    nop
    sbi   portd, 00
    mov   temp,  cnt1d
    andi  temp,  $f0        ;显示十位
    swap  temp
    rcall a
    cbi   portd, 01
    nop
    sbi   portd, 01
    mov   temp,  cnt2d
    andi  temp,  $0f        ;显示百位
    rcall a
    cbi   portd, 02
    nop
    sbi   portd, 02
    mov   temp,  cnt2d
    andi  temp,  $f0        ;显示千位
    rcall a
    swap  temp
    rcall a
    cbi   portd, 03
    nop
    sbi   portd, 03
    mov   temp,  cnt3d
    andi  temp,  $0f        ;显示万位
```

```
    rcall a
    cbi   portd, 04
    nop
    sbi   portd, 04
    mov   temp,  cnt3d
    andi  temp,  $f0        ;显示十万位
    swap  temp
    rcall a
    cbi   portd, 05
    nop
    sbi   portd, 05
    dec   cntn1
    brne  end2
    dec   cntn
    brne  end1
    ldi   tempn1,$01
    ret
a:
    ldi   zh,    high(zk*2)
    ldi   zl,    low(zk*2)
    add   zl,    temp
    lpm
    out   portb, r0
    ret
adjn:
    push  cntn
    mov   cntn,  cnt1d
    rcall adjd1
    mov   cnt1d, cntn
    mov   cntn,  cnt2d
    rcall adjd1
    mov   cnt2d, cntn
    mov   cntn,  cnt3d
    rcall adjd1
    mov   cnt3d, cntn
    mov   cntn,  cnt4d
    rcall adjd1
    mov   cnt4d, cntn
    mov   cntn,  cnt5d
    rcall adjd1
    mov   cnt5d, cntn
    pop   cntn
    ret
adjd1:
```

```
    ldi   tempn, 3
    add   tempn, cntn
    sbrc  tempn, 3
    mov   cntn,  tempn
    ldi   tempn, $30
    add   tempn, cntn
    sbrc  tempn, 7
    mov   cntn,  tempn
    wdr
    ret
.equ    zk=0x0200
.org    zk                          ;字形表
.db     0x03f,0x006,0x05b,0x04f
.db     0x066,0x06d,0x07d,0x007
.db     0x07f,0x06f,0x077,0x07c
.db     0x039,0x05e,0x071
```

7.4.4 LCD 显示 8 字循环

源程序:SLAVR744.ASM

LCD 显示器 1602AT(S)R 简介:

LED 显示器模块 1602AT(S)R 为 2X16 字符。含有 5*10 或 5*7 点 LCD 共 12*16=192 种 CG 显示字形及双组 8 个自由利用软件设定(CGRAM)的 5*8 图点字形,因此除内部固定 192 种字形外,再加上此 16 个可自由设定图字型等共计 208 种字图形如字符代码表所示。因 5*8 个点输入设定故 5 个点仅占用 D4-D0 的 5 位,而 D7-D5 则可为任意值,第八行值为游标地址,因此共八行占八个地址组成一个字形及标示游标地址,总共八个设定字图形,因此占有 $8*8=2^6$ 个地址,CG 地址设定值为 D5-D0。

LCD 引脚功能说明

1. GND:电源地,0V;
2. VCC 电源+5V;
3. VLC:LCD 驱动电压 0V-5V 对比度调节电压;
4. RS 寄存器选择信号;

   RS=0:

   指令寄存器 IR 写入(WRITE);

   (1) 忙(BUSY FLAG)读取(READ);

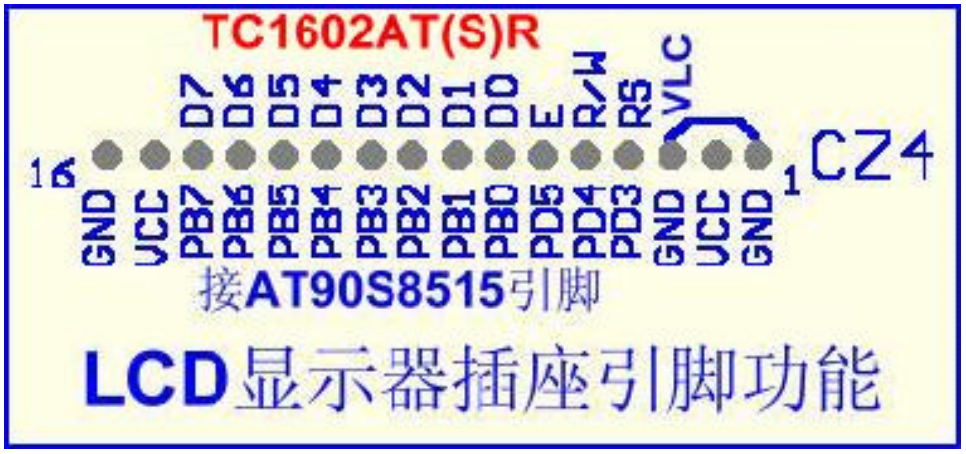
   (2) 地址计数器(ADDRESS COUNTER)AC 读取(READ);

   RS=1:数据寄存器(DATA REGISTER)读取及写入(READ/WRITE);
5. R/W 读/写控制信号(READ/WRITE):R/W=1 读取(READ)、R/W=0 写入(WRITE);
6. E(ENABLE)片使能信号,作写数据控制,下降沿触发;

7~14 脚为 DB0~DB7 八位数据总线,三态双向,若作为 4 位传送时应令:

DL=0,以 DB4-DB7 作传送将 8 位数据分二次传送;

15. 一般不用(空),如有背光 LED,则接 VCC;
16. 一般不用(空),如有背光 LED,则接 GND;



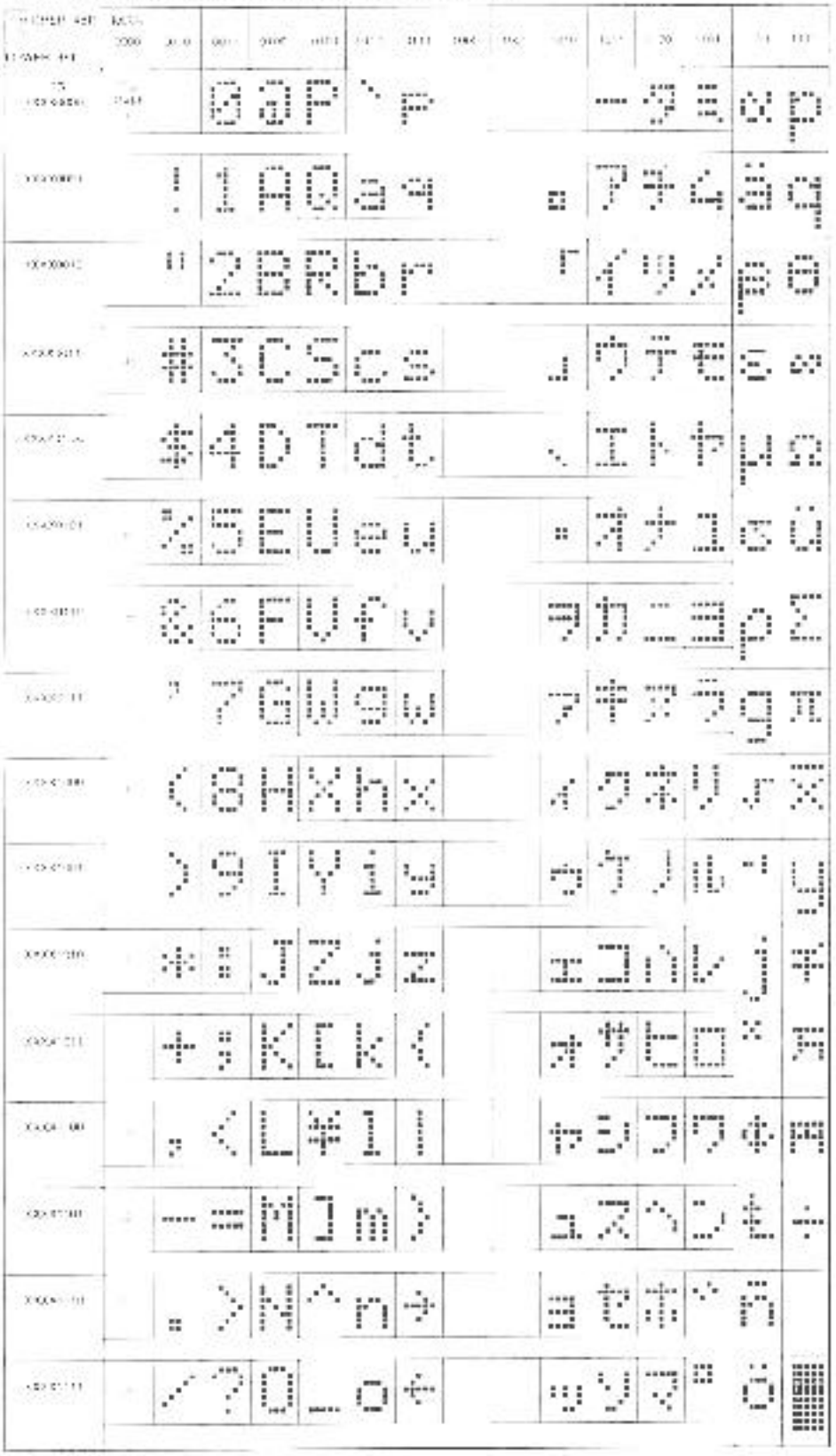
LCDTC1602 CG RAM **字形结构设定输入表**

字形码(DD RAM **数据**									CG RAM **地址**							**字形图样**(CG RAM)**数据**							
7	6	5	4	3	2	1	0		5	4	3	2	1	0		7	6	5	4	3	2	1	0
高位							**低位**		**上位**					**下位**		**上位**			(5*7 **字形**)				**下位**
												0	0	0		*	*	*	1	1	1	1	0
												0	0	1					1	0	0	0	1
												0	1	0					1	0	0	0	1
												0	1	1					1	1	1	1	0
0	0	0	0	*	0	0	0		0	0	0	1	0	0					1	0	1	0	0
												1	0	1					1	0	0	1	0
												1	1	0					1	0	0	0	1
												1	1	1		*	*	*	0	0	0	0	0
												0	0	0		*	*	*	1	0	0	0	1
												0	0	1					0	1	0	1	0
												0	1	0					1	1	1	1	1
												0	1	1					0	0	1	0	0
0	0	0	0	*	0	0	1		0	0	1	1	0	0					1	1	1	1	1
												1	0	1					0	0	1	0	0
												1	1	0					0	0	1	0	0
												1	1	1		*	*	*	0	0	0	0	0
												0	0	0		*	*	*					
												0	0	1									
0	0	0	0	*	1	1	1		1	1	1												
												1	0	1									
												1	1	0									
												1	1	1									

LCD 指令表

指 令	指 令 码										说 明	执行周期 f_{osc}= 250Khz
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0		
清屏	0	0	0	0	0	0	0	0	0	1	请除屏幕,置 AC 为 0, 光标回位	1.64ms
光标返回	0	0	0	0	0	0	0	0	1	*	DDRAM 地址为 0,显示回原位,DDRAM 内容不变	1.64ms
设置输入方法	0	0	0	0	0	0	0	1	I/D	S	设置光标移动方向并指定显示是否移动	40μs
显示开关	0	0	0	0	0	0	1	D	C	B	设置显示开或关(D),光标开关(C),光标所在字符闪烁(B)	40μs
移位	0	0	0	0	0	1	S/C	R/L	*	*	移动光标及整体显示,同时不改变 DDRAM 内容	40μs
功能设置	0	0	0	0	1	DL	N	F	*	*	设置接口数据(DL)、显示行数(L)、字符字体(F)	40μs
CGRAM 地址设置	0	0	0	1	ACG						设置 CGRAM 地址,设置后发送接收数据	40μs
DDRAM 地址设置	0	0	1	ADD							设置 DDRAM 地址,设置后发送接收数据	40μs
忙标志/读地址计数器	0	1	BF	AC							读忙标志(BF)标志正在执行内部操作并读地址计数器内容	40μs
CGRAM/DDRAM 数据写	1	0	写数据								从 CGRAM 或 DDRAM 写数据	40μs
CGRAM/DDRAM 数据读	1	1	读数据								从 CGRAM 或 DDRAM 读数据	40μs
	I/D=1:增量方式; I/D=0:减量方式 S=1:移位 S/C=1:显示移位; S/C=0 光标移位 R/L=1:右移; R/L=0 左移 DL=1:8 位; DL=0:4 位 N=1:2 行; N=0:1 行 F=1:5*10 字体; F=0:5*7 字体 BF=1:执行内部操作; BF=0:可接收指令										DDRAM:显示数据 RAM CGRAM:字符发生器 RAM ACG:CGRAM 地址 ADD:DDRAM 地址及光标地址 AC: 地址计数器用于 DDRAM 和 CGRAM	执行周期随主频改变而改变例如当 fcp 或 fosc=270 khz 时: 40μs x 250/270= 37μs

字符点阵LCD模块字符代码表



源程序:SLAVR744.ASM

```
.include"8515def.inc"
.def temp=r16
.def temp1=r17
.def temp2=r18
.def cnt=r20
.def cnt1=r21
.org $0000
    rjmp reset
.org $0030
reset:  ldi temp,low(ramend);设置堆栈指针。
    out spl,temp
    ldi temp,high(ramend)
    out sph,temp
    ldi temp,$ff            ;设置 D 口输出，B 口作输入。
    out ddrd,temp
    out portd,temp
    clr temp
    out ddrb,temp
    out portb,temp
    rcall syset             ;调用系统设置。
lp8:clr cnt1                ;循环程序。
lp81:   clr cnt
    ldi temp1,$80           ;设置第一行显示寄存器起址。(第二行为$a8)
    rcall contd
lp82:   cp cnt1,cnt
    brne lp83

    ldi temp1,$38       ;字形 8 的代码为$38。
lp84:   rcall writd
    inc cnt
    cpi cnt,$10
    brne lp82
    ldi temp,$55            ;设置延时常数。
    rcall delay
    inc cnt1
    cpi cnt1,$10
    brne lp81
    rjmp lp8
lp83:   ldi temp1,$20
    rjmp lp84

CONTD:   LDI    TEMP,0B00110000   ;写控制字入 LCD 中
     OUT    PORTD,TEMP
```

```
      RCALL   DELT3
      CBI     PORTD,$05          ;使 E=0 ，LCD 片选有效
      RCALL   DELT3
      SBI     PORTD,$05
BUSYY:    WDR
      SBIC    PINB,$07           ;读取 DB7=PINB7 是否为 0，为 0 则非忙跳过一行
      RJMP    BUSYY              ;DB7=1 为忙，跳回 BUSYY 再等待 DB7=0 以写入
      LDI     TEMP,0b00100000    ;写入数据写入控制字
      OUT     PORTD,TEMP
      RCALL DELT3                ;延时以免 AVR 速度太快而使 LCD 无法工作
      LDI     TEMP,$ff           ;设定 B 口为输出
      OUT     DDRB,TEMP
      OUT     PORTB,TEMP1        ;要写入 LCD 的数据 TEMP1 输出到 PORTB
      WDR
      CBI     PORTD,$05
      RCALL DELT3
      LDI     TEMP,0B00111000
      OUT     PORTD,TEMP
      CLR     TEMP
      OUT     DDRB,TEMP
      OUT     PORTB,TEMP
      RET

WRITD:    LDI TEMP,0B00110000      ;写数据入 LCD 中
      OUT PORTD,TEMP
      RCALL DELT3                ;延时以免 AVR 速度太快而使 LCD 无法工作
      CBI PORTD,$05              ;使 E=0，  LCD 片选有效
      RCALL DELT3
      SBI PORTD,$05
BUZY1:    WDR
      SBIC PINB,$07              ;读取 DB7=PINB7 是否为 0，为 0 则非忙跳过一行
      RJMP BUZY1                 ;DB7=1 为忙，跳回 BUSY1 再等待 DB7=0 以写入
      LDI TEMP,0B00101000      ;写控制字入 LCD 中
      OUT PORTD,TEMP
      OUT PORTB,TEMP1            ;要写入 LCD 的数据 TEMP1 输出到 PORTB
      LDI TEMP,$ff             ;设定 B 口为输入
      OUT DDRB,TEMP
      CBI PORTD,$05              ;使 E=0，  LCD 片选有效
      RCALL DELT3                ;延时以免 AVR 速度太快而使 LCD 无法工作
      LDI TEMP,0B00111000        ;写控制字入 LCD 中
      OUT PORTD,TEMP
      RCALL DELT3
      CLR TEMP                   ;PORTB 为输入
      OUT DDRB,TEMP
```

```
     OUT PORTB,TEMP          ;PORTB 为三态输入
     RET
syset:  ldi temp1,$01        ;清屏设定
    rcall contd
    ldi temp,$50             ;设置时间常数
    rcall delay
    ldi temp1,$38            ;2 行 5*7 显示设定
    rcall contd
    ldi temp1,$06            ;自动增量，显示不移位
    rcall contd
    ldi temp1,$0c            ;字形开关 ON，光标开关 OFF
    rcall contd
    ret
DELT3:  ldi temp2,$24
DT111:  wdr
    dec temp2
    brne dt111
    ret
delay:              ;延时子程序略
```

7.4.5 LED 电脑时钟

源程序:SLAVR745.ASM

硬件连接见:3.3 AVR 单片机开发下载实验器;

本程序若直接按 shife+exec 即从 00:00:00 开始计时。

本程序下载后(或上电后)，LED 显示 00:00:00 等待设置,请您从键盘上输入时、分、秒,要求输入位由小数点作光标提示,输入正确时间后,按执行键(SHIFT+EXEC)执行,电脑钟开始计时。

1. 请你如何修改程序,可当秒表用?
2. 又如何修改程序到点发出报时声?

```
  ;本程序在 SL-AVR 编程开发实验器上通过,由六位 LED 显示
  ;本程序采用 T0，1/1024 分频，设定一次中断为 25MS,40 次中断为 1 秒。
  .include"8515def.inc"
 .def   TEMP  =r16
 .def   TEMP1 =r17
 .def   temp2 =r18
 .def   temp3 =r19
 .def   CNT   =r20
 .def   SCNN  =r21
 .def   KSNI  =r22
 .def   SCNDP =r23
 .def   KEYN  =r24
 .def   cnt1 =r25
 .def   hour=r24
 .def   minute=r22
```

```
 .def   second=r21
 .equ label=$0f00        ;字形表首址
.org $0000
    rjmp reset
.org $007
intt0:  ldi temp,104     ;因 25ms 内差 40us 故补上 40/(1/8)即 320 个 CK。
bu: dec temp             ;因中断需 4CK 这样:4+104*(1+2)+1+1+1+1=320。
    brne bu
    nop
    inc cnt1             ;cnt1 计数 40 次为 1 秒钟。
    ldi temp,256-195     ;计数(256-195)次才产生 1 次中断。
    out tcnt0,temp       ;CK/1024 分频,这样一次中断需 25ms。
    rjmp recog
.org $030
reset:
    ldi temp,low(ramend) ;设置堆栈指针。
    out spl,temp
    ldi temp,high(ramend)
    out sph,temp
    clr r2           ;清工作寄存器。
    clr r3
    clr r4
    clr r5
    clr r6
    clr r7
    clr zh
    clr xh
    clr yh
    clr keyn
    clr second
    clr minute
    clr hour
    clr cnt
    clr yh
    ldi temp,$80
    mov r8,temp      ;R8=$80
    ldi yl,$60       ;设置显示内存地址指针 Y 为$0060.
    rcall disram     ;调用 DISRAM。
    ld temp,y
    ldi temp1,$80
    add temp,temp1
    st y,temp
scanad: ldi temp,$07
    ldi yl,$60
```

```
 scann: rcall scan1      ;调用键扫显示子程序 SCAN1。
    brts scann
 scank: rcall scan1
    brtc scank
    rcall scan1
 scans: nop
        cpi keyn,$10     ;KEYN=$10 转 EXEC。
    brcc exec
    rcall wraddram       ;调用 WRADDRAM。
    dec temp     ;TEMP 减 1。
    cpi temp,$01
    brne scann       ;TEMP=1 则转 SCANN
    rjmp scanad
exec:   mov temp1,r7;把 r7,r6 的两个十进制换成一个十六进制入 hour 中
    mov temp,r6
    rcall dechex
    mov hour,temp
    mov temp1,r5;把 r5,r4 的两个十进制换成一个十六进制入 minute 中
    mov temp,r4
    rcall dechex
    mov minute,temp
    mov temp1,r3;把 r3,r2 的两个十进制换成一个十六进制入 second 中
    mov temp,r2
    rcall dechex
    mov second,temp
    ldi temp,$05     ;T0 设置为 CK/1024 分频。
    out tccr0,temp
    ldi temp,256-195
    out tcnt0,temp       ;装载 T0 时间常数。
    ldi temp,$ff     ;设置 b 口,d 口为输出
    out ddrb,temp
    out ddrd,temp
    sei          ;开中断总开关
    ldi temp,$02
    out timsk,temp       ;允许 t0 中断。
display:rcall disram     ;调用 disram
    clr yh           ;设置显示内存地址指针 Y 为$0060

    ldi yl,$60
    ldi scndp,$df        ;设置扫描显示码 SCNDP 起址 0B11011111.
agdis:  ld r1,y+
    cpi yl,$62
    brne npoint
    add r1,r8
```

```
npoint: cpi yl,$64
    brne next
    add r1,r8
next:   out portb,r1     ;把 R1 送 B 口显示
    out portd,scndp      ;扫亮某个数码管
    sec          ;C=1
    ror scndp        ;右移 SCNDP
    rcall delay      ;延时
    cpi yl,$66
    brne agdis       ;未扫亮最后一位继续
    rjmp display
recog:  cpi cnt1,40      ;40 次中断为 40*25ms=1 秒
    brne inthome     ;40 次中断未到转 inthome
    clr cnt1     ;40 次中断到则清 cnt1
    inc second       ;秒寄存器加 1
    cpi second,60
    brne change      ;秒寄存器未满转 change
    clr second       ;否则清秒寄存器
    inc minute       ;分寄存器加 1
    cpi minute,60
    brne change      ;分寄存器未满转 change
    clr minute       ;否则清分寄存器
    inc hour     ;时寄存器加 1
    cpi hour,24
    brne change      ;时寄存器未满转 change
    clr hour     ;否则清时寄存器
    reti         ;中断返回
change: mov temp,second  ;把 second 中的十六进制转换成二个十进制数存入 r3,r2 中
    rcall hexdec
    mov r3,temp1
    mov r2,temp
    mov temp,minute ;把 minute 中的十六进制转换成二个十进制数存入 r5,r4 中
    rcall hexdec
    mov r5,temp1
    mov r4,temp
    mov temp,hour    ;把 hour 中的十六进制转换成二个十进制数存入 r7,r6 中
    rcall hexdec
    mov r7,temp1
    mov r6,temp
inthome:reti     ;中断返回
hexdec: clr temp1      ;把 temp 中的十六进制转成二个十进制入 temp1,temp 中的子程序
hexdec1:subi temp,10
    brcs negs
    inc temp1
```

```
    rjmp hexdec1
negs:   subi temp,$f6
    ret             ;子程序返回
dechex: push temp   ;把 temp1,temp 的两个十进制数转换成一个十六进制入 temp 中
    ldi temp2,$0a
    clr temp
dechex1:cpi temp1,$00
    breq dh
    dec temp1
    add temp,temp2
    rjmp dechex1
dh: pop temp1
    add temp,temp1
    ret         ;子程序返回
disram: push yl     ;压栈保护
    push zl
    push xl
    ldi zh,high(label*2) ;Z 指针指向字形表首址 label*2
    ldi zl,low(label*2)
    clr xh
    ldi xl,$60
    ldi yl,$07
ramag:  ld temp2,y      ;y 为间址的内容送 temp2
    dec yl
    mov zl,temp2
    lpm
    st x+,r0    ;把 r0 的内容送到$0060-$0065 中
    cpi xl,$66
    brne ramag
    pop xl
    pop zl
    pop yl          ;退栈
    ret     ;子程序返回
wraddram:push temp      ;读键存入显示内存及寄存器中。
    clr zh
    mov zl,temp
    st z,keyn
    ldi zh,high(label*2)
    mov zl,keyn
    lpm
    st y+,r0
    cpi yl,$66
    brne pointc
    ldi yl,$60
```

```
pointc: ld temp2,y
    ldi temp3,$80
    add temp2,temp3
    st y,temp2
    pop temp
    ret     ;子程序返回
delay:  push temp    ;延时子程序
lp1:ldi temp2,$10
lp2:dec temp
    brne lp2
    dec temp2
    brne lp2
    pop temp
    ret     ;子程序返回
 SCAN1:         push xh          ;键盘扫描显示子程序(注释从略,见 7.3.3)。
        PUSH XL
        PUSH TEMP1
        PUSH TEMP
        LDI XL,$60
            SET
            LDI  SCNN,$00
            LDI  SCNDP,0B11011111
            LDI  CNT,$06
            LDI  KSNI,0B11110111
      COL1: LDI  TEMP,$FF
            OUT  DDRb,TEMP
            OUT  DDRC,TEMP
        OUT  PORTC,TEMP
            OUT  DDRd,TEMP
        OUT  PORTd,SCNDP
            LD   R1,X+
            OUT  PORTb,R1
            RCALL  DELAY
            MOV  TEMP,CNT
            SUBI  TEMP,$03
            BRCS  NOSK
            LDI  TEMP1,$04
            LDI  TEMP,0B00001111
            OUT  DDRc,TEMP
            OUT  PORTc,KSNI
            RCALL  DELYT
            IN   TEMP,PINc
            ANDI  TEMP,0B11110000
            SWAP  TEMP
```

```
          KROW: SEC
        ROR  TEMP
            BRCS  NOKEY
            CLT
            MOV   KEYN,SCNN
        SBIS PINd,$07
        ADIW KEYN,$10
         NOKEY: INC  SCNN
            DEC  TEMP1
            BRNE  KROW
            SEC
            ROR  KSNI
          NOSK: SEC
            ROR  SCNDP
            DEC  CNT
            BRNE  COL1
            LDI   TEMP,$FF
            OUT   DDRC,TEMP
        OUT   PORTC,TEMP
        POP TEMP
        POP TEMP1
        POP XL
        pop xh
        RET
         delyt: ldi temp3,$20
       dt31:dec temp3
        brne dt31
        ret

.cseg
.org $0f00              ;字形表
 .dw 0x063f,0x4f5b,0x6d66,0x077d
 .dw 0x6f7f,0x7c77,0x5e39,0x7179
```

7.4.6 测频率

源程序:SLAVR746.ASM

```
;测频率,信号从 AT90S8515 的 ICP 引脚输入,最大值为为 999999Hz/μS
;本程序在 SL-AVR 编程开发实验器上通过,由六位 LED 显示
.include"8515def.inc"
    rjmp   reset
.def    temp   = r16                ;暂存器

.def    cnt1d  = r17
.def    cnt2d  = r18                ;cnt1、dcnt2d 和 cnt3d 存放结果的十进制
.def    cnt3d  = r19

.def    count  = r20

.def    cnt    = r21

.def    res1   = r22
.def    res2   = r23                ;res1、res2 和 res3 存放结果的十六进制
.def    res3   = r24

.def    dt     = r25
.def    ovfl   = r26
.def    aa     = r27
.org    0x003                       ;icp 触发中断向量
    rjmp   captr

.org    0x007                       ;timer0 触发中断向量
    rjmp   interru
captr:                              ;icp 触发中断子程序
    brts  b
    inc   res1
    ldi   temp,  0b00000101
    out   tccr0, temp           ;开 timer0
    ldi   temp,  0b00001010
    out   timsk, temp           ;致 timer0 中断和捕捉中断
    ldi   temp,  0b11000000
    out   tccr1b,temp
    set
    reti
b:
    set
    inc   res1                  ;开始计数
    brne  c
```

```
    inc   res2
    brne  c
    inc   res3
    cpse  res3,  ovfl            ;溢出处理
    rjmp  c
    rjmp  over1
c:
    ldi   temp,  0b00001010
    out   timsk, temp
    ldi   temp,  0b11000000
    out   tccr1b,temp
    reti

interru:                                ;timer0 溢出中断子程序
    dec   cnt
    breq  over
    ldi   temp,   0b00001010
    out   timsk,  temp
    reti
over:
    rcall htd3
over1:
    rcall sys
    reti
reset:
    ldi   temp,  low(ramend)
    out   spl,   temp
    ldi   temp,  high(ramend)       ;设置堆栈
    out   spl+1, temp
    ldi   temp,  $ff                ;初始化数码管状态
    out   ddrb,  temp               ;B 口:数码管数据输出
    out   ddrd,  temp               ;D 口:pd0-pd5 为数码管片选
    ldi   temp,  $00
    out   portb, temp               ;共阴极,数码管全灭
    out   portd, temp
    ldi   cnt1d, 00
    ldi   cnt2d, 00
    ldi   cnt3d, 00
    sei
    rcall sys
loop:                                   ;在数码管显出十进制数
    mov   aa,   cnt1d
    andi  aa,   $0f                 ;显示个位
    rcall a
```

```
    cbi   portd, 00
    nop
    sbi   portd, 00
    mov   aa,  cnt1d
    andi  aa,  $f0              ;显示十位
    swap  aa
    rcall a
    cbi   portd, 01
    nop
    sbi   portd, 01
    mov   aa,  cnt2d
    andi  aa,  $0f              ;显示百位
    rcall a
    cbi   portd, 02
    nop
    sbi   portd, 02
    mov   aa,  cnt2d
    andi  aa,  $f0              ;显示千位
    swap  aa
    rcall a
    cbi   portd, 03
    nop
    sbi   portd, 03
    mov   aa,  cnt3d
    andi  aa,  $0f              ;显示万位
    rcall a
    cbi   portd, 04
    nop
    sbi   portd, 04
    mov   aa,  cnt3d
    andi  aa,  $f0              ;显示十万位
    swap  aa
    rcall a
    cbi   portd, 05
    nop
    sbi   portd, 05
    sbrc  dt,    0
    ret
    rjmp  loop
sys:                                ;初始化
    clc
    clt
    ldi   dt,    00
    ldi   ovfl,  $0f
```

```
    ldi   cnt,   31
    ldi   temp,  123
    out   tcnt0, temp
    ldi   temp,  0b00001000
    out   timsk, temp
    ldi   temp,  0b11000000
    out   tccr1b,temp
    ret

htd3:                                   ;16 转 10 子程序
    sbr   dt,    1
    ldi   count, 24
    clr   cnt1d
    clr   cnt2d
    clr   cnt3d
loopd:
    rol   res1
    rol   res2
    rol   res3

    rol   cnt1d
    rol   cnt2d
    rol   cnt3d
    dec   count
    brne  d
    rjmp  loop
d:
    rcall adjn
    rjmp  loopd
a:
    ldi   zh,    high(zk*2)
    ldi   zl,    low(zk*2)
    add   zl,    aa
    lpm
    out   portb, r0
    ret
adjn:
    push  count
    mov   count,  cnt1d
    rcall adjd1
    mov   cnt1d,  count
    mov   count,  cnt2d
    rcall adjd1
    mov   cnt2d,  count
```

```
    mov   count,  cnt3d
    rcall adjd1
    mov   cnt3d,  count
    pop   count
    ret
adjd1:
    ldi   temp,  3
    add   temp,  count
    sbrc  temp,  3
    mov   count, temp
    ldi   temp,  $30
    add   temp,  count
    sbrc  temp,  7
    mov   count, temp
    ret
.equ   zk=0x0200
.org   zk                              ;字形表
.db    0x03f,0x006,0x05b,0x04f
.db    0x066,0x06d,0x07d,0x007
.db    0x07f,0x06f,0x077,0x07c
.db    0x039,0x05e,0x071,0x0ff
```

7.4.7 测转速

源程序:SLAVR747.ASM

```
; 测转速,信号从 AT90S8515 的 ICP 引脚输入,最大值为为 999999 转/分
;本程序在 SL-AVR 编程开发实验器上通过,由六位 LED 显示
.include"8515def.inc"
    rjmp reset
.def    temp  = r16              ;暂存器
.def    aa    = r17
.def    cnt   = r18
.def    mc16l = r19          ;mc16l 和 mc16h 存放脉冲个数
.def    mc16h = r20
.def    mp8u  = r21                  ;mp8u=30,因为是每 2 秒采样
.def    res1  = r21                  ;res1、res2 和 res3 存放结果的十六进制
.def    res2  = r22
.def    res3  = r23
.def    count = r24
.def    cnt1d = r25                  ;cnt1、dcnt2d 和 cnt3d 存放结果的十进制
.def    cnt2d = r26
.def    cnt3d = r27
.def    dt    = r28
```

```
.def    dt1   = r29
.org 0x003                          ;icp 触发中断向量
icpt1:
    rjmp  captr
.org 0x008
    rjmp  interru
.cseg
.org 0x010
captr:                                  ;icp 触发中断子程序
    brts  down
    ldi   temp,  0b00000101
    out   tccr0, temp           ;开 timer0
    ldi   temp,  0b00001010
    out   timsk, temp           ;致 timer0 中断和捕捉中断
    ldi   temp,  0b10000000
    out   tccr1b,temp
    set
    reti
down:                                   ;下降沿开始计数
    set
    inc   mc16l
    brne  b
    inc   mc16h
    cpse  mc16h, dt1            ;溢出处理
    rjmp  b
    ldi   mc16h, 00
    ldi   mc16l, 00
    rjmp  over
b:
    ldi   temp,  0b00001010     ;致 timer0 中断和捕捉中断
    out   timsk, temp
    ldi   temp,  0b10000000
    out   tccr1b,temp
    reti
interru:                                ;timer0 溢出中断子程序
    dec   cnt
    breq  over
    ldi   temp,  0b00001010
    out   timsk, temp
    reti
over:
    rcall conver                ;conver:计算结果子程序
    rcall htd3                  ;htd3:
    rcall sys
```

```
    reti

reset:
    ldi   temp,  low(ramend)
    out   spl,   temp
    ldi   temp,  high(ramend)      ;设置堆栈
    out   spl+1, temp
    ldi   temp,  $ff               ;初始化数码管状态
    out   ddrb,  temp              ;B 口:数码管数据输出
    out   ddrd,  temp              ;D 口:pd0-pd5 为数码管片选
    ldi   temp,  $00
    out   portb, temp              ;共阴极,数码管全灭
    out   portd, temp
    ldi   cnt1d, 00
    ldi   cnt2d, 00
    ldi   cnt3d, 00
    sei
    rcall sys
loop:                                  ;在数码管显出十进制数
    mov   aa,   cnt1d
    andi  aa,   $0f                ;显示个位
    rcall a
    cbi   portd, 00
    nop
    sbi   portd, 00
    mov   aa,  cnt1d
    andi  aa,  $f0                 ;显示十位
    swap  aa
    rcall a
    cbi   portd, 01
    nop
    sbi   portd, 01
    mov   aa,  cnt2d
    andi  aa,  $0f                 ;显示百位
    rcall a
    cbi   portd, 02
    nop
    sbi   portd, 02
    mov   aa,  cnt2d
    andi  aa,  $f0                 ;显示千位
    swap  aa
    rcall a
    cbi   portd, 03
    nop
```

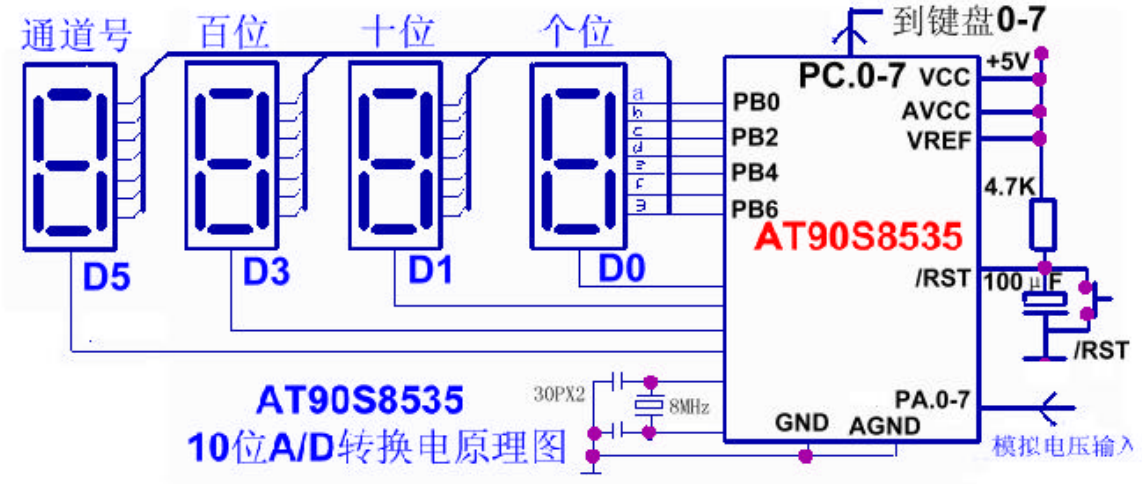
```
    sbi   portd, 03
    mov   aa,  cnt3d
    andi  aa,  $0f                ;显示万位
    rcall a
    cbi   portd, 04
    nop
    sbi   portd, 04
    mov   aa,  cnt3d
    andi  aa,  $f0                ;显示十万位
    swap  aa
    rcall a
    cbi   portd, 05
    nop
    sbi   portd, 05
    sbrc  dt,    0
    ret
    rjmp  loop
sys:                              ;初始化
    clc
    clt
    ldi   dt1,   $81
    ldi   dt,    00
    ldi   mc16l, 00
    ldi   mc16h, 00
    ldi   cnt,   62
    ldi   mp8u,  30
    ldi   temp,  247
    out   tcnt0, temp
    ldi   temp,  00
    out   tccr1a,temp
    out   tccr0, temp
    ldi   temp,  0b00001010
    out   timsk, temp
    ldi   temp,  0b11000000
    out   tccr1b,temp
    ret

htd3:                             ;16 转 10 子程序
    sbr   dt,    1
    ldi   count, 24
    clr   cnt1d
    clr   cnt2d
    clr   cnt3d
loopd:
```

```
    rol   res1
    rol   res2
    rol   res3

    rol   cnt1d
    rol   cnt2d
    rol   cnt3d
    dec   count
    brne  c
    rjmp  loop
c:
    rcall adjn
    rjmp  loopd
a:
    ldi   zh,    high(zk*2)
    ldi   zl,    low(zk*2)
    add   zl,    aa
    lpm
    out   portb, r0
    ret
adjn:
    push  count
    mov   count,  cnt1d
    rcall adjd1
    mov   cnt1d,  count
    mov   count,  cnt2d
    rcall adjd1
    mov   cnt2d,  count
    mov   count,  cnt3d
    rcall adjd1
    mov   cnt3d,  count
    pop   count
    ret
adjd1:
    ldi   temp,  3
    add   temp,  count
    sbrc  temp,  3
    mov   count, temp
    ldi   temp,  $30
    add   temp,  count
    sbrc  temp,  7
    mov   count, temp
    ret
conver:                          ;计算结果子程序
```

```
    clr   res3
    clr   res2
    ldi   cnt,   8
    lsr   mp8u
m16_1:
    brcc  m16_2
    add   res2,  mc16l
    adc   res3,  mc16h
m16_2:
    ror   res3
    ror   res2
    ror   res1
    dec   cnt
    brne  m16_1
    ret

.equ    zk=0x0200
.org    zk                                  ;字形表
.db     0x03f,0x006,0x05b,0x04f
.db     0x066,0x06d,0x07d,0x007
.db     0x07f,0x06f,0x077,0x07c
.db     0x039,0x05e,0x071
```

7.4.8 AT90S8535 的 A/D 转换

源程序:SLAVR748.ASM




```
;用 AT90S8535 作 0-7 通道 A/D 转换,用 LED 显示,左一位(D5)显示通道号,
;右三位(D2-D0)显示转换值(十六进制数 0-3FFH),程序下载即执行,
;自动从 0 通道到 7 通道 A/D 转换扫描显示,当你按下 0-7 任一位数字键,
```

```
;该通道显示时间延长一段时间,然后又自动循环显示。
;本程序在 SL-AVR 编程开发实验器上通过,由四位 LED 显示。
;硬件接口: AT90S8535 的 PB.0-7 接 LED 段显示(用短路块短接),PD.0-5 接 LED 位显示,用接插线
连接,
;PC0-PC7 接键盘线
;PA.0-7 接模拟电压,滑线电位器 A/D VX 端,
;AGND 接地
;最好 AVCC 与 VRBF 间接 1K 电阻,VRBF 到地接 100μF 电解电容,
;AVC 与 VCC 间接一只 100Ω电阻,AVCC 接 104 瓷片电容到地,
;/RST 接上复位按钮,插上 CZ2 到 AT4 下载线,即连通晶振引脚线,
.include  "8535def.inc"
 .org  $0000
     rjmp  reset
 .def   TEMP  =r16
 .def   TEMP1 =r17
 .def   temp2 =r18
 .def   temp3 =r19
 .def   CNT   =r20
 .def   scndp =r21
 .def   KSNI  =r22
 .def   SCNN  =r23
 .def   KEYN  =r24
 .def temp4=r25
 .equ   label =$0f00
 .org 0030
reset:  ldi temp,high(ramend);设置堆栈指针.
    out sph,temp
    ldi temp,low(ramend)
    out spl,temp
    clr xh       ;设置 x 指针为$0061.
    ldi xl,$61
    clr temp;清$0061,$0062 单元.
    st x+,temp
    st x,temp
init:   clr temp2    ;由 0 通道开始.
next:   ldi temp3,$01
next1:  clr temp4
again:  rcall cance ;调用 a/d 转换子程序 cance.
lp: rcall scan1 ;调用键扫显示子程序 scan1.
 scann: rcall scan1
    brtc recog  ;用按键转 recog.
    inc temp4   ;键扫显示次数 temp4 加 1.
    cpi temp4,$ff
    brne again  ;temp4 不等于$ff 转 again.
```

```
    dec temp3
    brne again  ;temp3 不等于 0 转 again.
    inc temp2   ;通道代码 temp2 加 1.
    cpi temp2,$08
    brne next   ;8 个通道未结束转下一通道 next.
    rjmp init   ;8 个通道已扫描完再重扫.
recog:  cpi keyn,$08
    brcc next   ;无效键转 next.
    ldi temp3,$04   ;设置有效通道键按下后的循环次数.
    mov temp2,keyn  ;通道数送 temp2.
    rjmp next1
cance:  mov temp,temp2  ;a/d 转换子程序.
    out admux,temp  ;设置通道.
    ldi temp,$86;设置 a/d 转换使能且采用 1/64 分频作转换工作频率.
    out adcsr,temp
    sbi adcsr,adsc  ;启动转换.
  loop: sbic adcsr,adsc ;转换结束跳行否则等待.
    rjmp loop
    in r2,adcl  ;把转换结果送 r2.r3.
    in r3,adch
    mov temp,temp2
    rcall wrdisram  ;调用把转换的结果转换成显示代码 wrdisram.
    ret     ;转换结束返回.
wrdisram:clr xh     ;使 x 指针为$0060.
    ldi xl,$60
    rcall fetch ;调用 fetch.
    st x+,temp  ;把 temp 存入$0060 单元.
    inc xl
    inc xl
    mov temp,r3
    andi temp,$0f   ;取 r3 的低 4 位.
    rcall fetch ;取字形代码.
    st x+,temp
    mov temp,r2
    swap temp
    andi temp,$0f   ;取 r2 的高 4 位.
    rcall fetch ;取字形代码.
    st x+,temp
    mov temp,r2
    andi temp,$0f   ;取 r2 的低 4 位.
    rcall fetch ;取字形代码.
    st x+,temp
    ret     ;返回.
fetch:  ldi zh,high(label*2);设置字形表指针 z.
```

```
    mov zl,temp
    lpm          ;取字形.
    mov temp,r0      ;字形码送 temp.
    ret          ;返回
 SCAN1:          push xh          ;键扫显示子程序。
        PUSH XL                   ;将 xl 压入堆栈
        PUSH TEMP3
        PUSH TEMP2
        PUSH TEMP1
        PUSH TEMP
        lDI XL,$60
            SET                   ;T 标志为 1 表示未按键
            LDI  SCNN,$00         ;按键起始扫描码 SCNN 为 00
            LDI  SCNDP,0B11011111 ;令 6 位七段 LED 扫描显示码初始为 11011111
            LDI  CNT,$06          ;七段 LED 共 6 位故 CNT=6 为位数计数
            LDI  KSNI,0B11110111  ;4*4 键盘扫描码 KSNI 初始为 11110111
      COL1: LDI  TEMP,$FF         ;PORTB 设定为输出
            OUT  DDRb,TEMP
            OUT  DDRC,TEMP        ;PORTC 设定为输出
        OUT  PORTC,TEMP
            OUT  DDRd,TEMP        ;PORTD 设定为输出
        OUT  PORTd,SCNDP        ;6 位七段 LED 扫描显示码输出到 PORTD
        CLR  XH
            LD   R1,X+            ;要显示于七段 LED 的间接寄存器 X 中的内容送入 R1 并
令 X 加 1
            OUT  PORTb,R1         ;显示内容输出到 PORTB 以驱动 LED 显示
            RCALL  DELAY          ;调用延时以显示此位数一段时间
            MOV  TEMP,CNT         ;LED 位数为 6 而按键码行数为 4 故需作 CNT 值检测
            SUBI  TEMP,$03        ;CNT=TEMP 与 3 相减比较
            BRCS  NOSK            ;位数扫描 CNT 超过 3 则 C 为 1 跳到 NOSK 不作按键处理
            LDI  TEMP1,$04        ;一共要检查 4 个按键
            LDI  TEMP,0B00001111  ;设定 PC0-PC3 为输出 PC4-PC7 为输入
            OUT  DDRc,TEMP
            OUT  PORTc,KSNI       ;KSNI 输出到 PORTC 并令 PC7-PC4 为上拉电阻输入态
            RCALL  DELYT          ;调用延时以稳定读取键盘 I/O 输入端
            IN   TEMP,PINc        ;读取 C 口检测 PC7-PC4 看是否有按键低电位输入
            ANDI  TEMP,0B11110000 ;取 TEMP 的高 4 位
            SWAP  TEMP            ;键码顺序为 PC4-PC7 故将 TEMP 的高低 4 位互换成 D0-D3
         KROW: SEC                  ;令 C 标志为 1 以便将键盘码 D0-D3 移到 C 标志位检
测
        ROR  TEMP               ;TEMP 的内容右移 1 位将第一个键码 D0=PC4 移到 C 标志位检
测
            BRCS  NOKEY           ;若有键按下则测到 PC4=D0=0，若 C=1 无按键则转到
NOKEY
```

```
            CLT                     ;若 PC4=D0=CF=0 表示有按键令 T=0 表示有按键
            MOV   KEYN,SCNN         ;把按键扫描码 SCNN 送键码 KEYN 中保存
        SBIS PINd,$07
        ADIW KEYN,$10           ;判定 SHIFT 键是否按下，按下则键值加 10
         NOKEY: INC  SCNN               ;按键扫描码 SCNN 加 1
            DEC  TEMP1          ;扫描读取键数 TEMP1 减 1
            BRNE  KROW          ;每行有 4 个按键如 TEMP1 不为 0 则跳到 KROW 再检测
PC5-PC7
            SEC                 ;此行 4 个键码检测完后令 C 为 1 以方便键盘扫描码 KSNI
内容的移位
            ROR  KSNI           ;键盘扫描码 KSNI=CF=1>11110111 移位以进行下一行按
键扫描
          NOSK: SEC                 ;令进位标志 CF=1
            ROR  SCNDP          ;将扫描显示码 SCNDP 左移作下一位扫描
            DEC  CNT            ;共需作 6 位数扫描显示故 CNT 减 1
            BRNE  COL1          ;CNT 减 1 不为 0 则跳回 COL1 再作扫描显示及读取键盘
输入
            LDI   TEMP,$FF      ;若已完成全部扫描显示和读取按键则令 TEMP=0ff
            OUT   DDRC,TEMP     ; TEMP 输出到 DDRC 设定 PORTC 为输出驱动 LED
        OUT   PORTC,TEMP
        POP TEMP
        POP TEMP1
        POP TEMP2
        POP TEMP3
        POP XL
        pop xh
        RET
      delay:push temp1
        push temp3
        ldi temp1,$10
       dt11:ldi temp3,$20
          dt21:nop
        dec temp3
        brne dt21
        dec temp1
        brne dt11
        pop temp3
        pop temp1
        ret
      delyt:ldi temp3,$20
       dt31:dec temp3
        brne dt31
        ret
.cseg
```

```
.org $0f00
 .dw 0x063f,0x4f5b,0x6d66,0x077d
 .dw 0x6f7f,0x7c77,0x5e39,0x7179
```