

本教程之版权归 APZ 所有，如需转载请保持文本完整。

参考书目：

1. Intel 80C31/80C51 Microcontroller Datasheet
2. 〈单片机应用文集 1〉 何立民主编

SCB-2 型 MCS-51 单片单板机的硬件结构如下：

CPU：8031。

EPROM：ROM 选用一片 16K EPROM27128，其中前 8K 放置 MCS-51 监控，后 8K 可放置一些应用程序。

RAM：外部随机存储器选用 2 片静态 8K RAM6264，共 16K。外部 RAM 配有掉电保护电路。在这 2 片 RAM 的插座上可以对 2864A 进行编程。

PIO：外部并行口选用一片 8155，利用它，可实现 EPROM 编程，此外还可以驱动打印机等。

A/D：外部配的 AD 是 ADC0809，它是 8 路 8 位 A/D。

D/A：外部配的 DA 是 DAC0832，它是 1 路 8 位 D/A。

键输入：本机配有 25 个按键，其中一个做复位键用，16 个做数值键，8 个做命令键。

显示输出：显示输出由 6 个共阴数码管组成，4 个用来显示地址或状态信息，2 个显示数据或代码。

RS232 接口：SCB-2 型单片单板机通过一片 1488 和一片 1489 形成简易的 RS232 接口，通过该接口与高位机进行通讯。

```

;=====
;*      初始化模块      START      *
;=====
; 功能:初始化堆栈指针,显示缓冲区. 监控标志单元和监控所用的片
;      内 RAM 其它单元.
; 入口:上电或复位后即进入该模式.
; 出口:完成初始化工作后进入主控模块.

        ORG 0000H
        NOP
        AJMP START
        CLR EX0 ;禁止外部中断 0
        LJMP PUSH_REG ;单步中断,到保护片内 RAM

        ORG 000BH ;中断向量入口地址,定时器 0 溢出
        LJMP 200BH
        NOP
        NOP
        ORG 0013H ;外部中断 1
        LJMP 2013H
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
```

```

NOP
LJMP 201BH ;定时器 1 溢出
NOP
NOP
NOP
NOP
NOP
LJMP 2023H ;串行口
NOP
NOP
NOP
ORG 0030H
START:  MOV R0,#79H ;初始化显示缓冲区
        MOV @R0,#01H ;"1"
        INC R0
        MOV @R0,#05H ;"5"
        INC R0
        MOV @R0,#14H ;"-"
        INC R0
        MOV @R0,#1CH ;"U"
        INC R0
        MOV @R0,#10H ;"P"
        INC R0
        MOV @R0,#0CH ;"C"
        MOV SP,#40H ;堆栈
        MOV A,#00H
        MOV 2FH,A ;清除监控标志单元
        MOV R0,#30H
        MOV R1,#10H
START0:  MOV @R0,A
        INC R0
        DJNZ R1,START0

;=====
;*      主控模块  MONITOR      *
;=====
; 功能:维持显示并等待键输入;若按下的为数字键,则将高 4 位显示器
;      内容左移一位,将按下的数字键填入空出的显示器, 然后继续
;      等待下一按键;若按下的为功能键,则散转到各功能键入口.
; 入口:无任何要求,在初始化模块之后执行.
; 出口:到相应的功能键处理模块.
        ORG 0052H
MONITOR:  LCALL KEY_SCAN ;调键扫描
          JNZ KEY_PROCESS ;有键按下转处理
          LCALL LED_DISPLAY ;无键按下调显示

```

```

                                AJMP MONITOR
KEY_PROCESS:                    LCALL OFF_DISP ;关闭显示
K_P1:                          LCALL KEY_GET ;求出键号
                                JB ACC.4,COMMAND
                                MOV 7EH,7DH ;数字键处理
                                MOV 7DH,7CH
                                MOV 7CH,7BH
                                MOV 7BH,A
                                AJMP K_P1
COMMAND:                        ANL A,#07H ;命令键处理
                                MOV R0,A
                                RL A
                                ADD A,R0
                                MOV DPTR,#FUNCTION
                                JMP @A+DPTR ;功能键散转
FUNCTION:                       LJMP WRIT
                                LJMP F
                                LJMP MOV
                                LJMP UP
                                LJMP DOWN
                                LJMP EXEC
                                LJMP READ
                                LJMP EXAM
;=====
;*      键处理模块      EXAM      *
;=====
; 功能:检查存储器或片内 RAM 指定单元的内容,若本单元内容已显示,
;      则显示下一单元地址及其内容;然后等待按键,若为数字键,则
;      低 2 位显示器内容左移一位, 键值送最低位, 然后继续等待按
;      键;若为功能键,则跳到主控模块的功能处理部分.
; 入口:要求高 4 位显示缓冲区中为指定单元的地址,存储器用 4 位表示
;      片内 RAM 用其中的低 2 位表示.
; 出口:得到功能键后跳到主控模块的功能键处理部分;不满足入口要
;      求时显示错误信息后返回主控模块.
                                ORG 00E6H
EXAM:                          JNB 78H,EXAMD
                                LJMP EPROM ;上档功能操作,到 EPROM 固化
EXAMD:                         CLR 7CH ;消除错误标志位
                                ACALL EXAMS
                                JNB 7CH,EXAMC ;未出错,即是按下功能键返回
                                LJMP MONITOR
EXAMC:                         LJMP COMMAND ;到主控模块功能键处理部分
EXAMS:                         MOV A,7EH
                                JB ACC.4,EXAM1

```

```

                                AJMP EXAM_MEM
EXAM1:                        MOV A,7DH ;片内 RAM 单元映像区检查
                                JB ACC.4,EXAM2
                                AJMP EXAM_ERR ;非法 RAM 地址,出错
EXAM2:                        MOV A,7CH
                                JNB ACC.4,EXAM_REG
                                AJMP EXAM_ERR ;非法 RAM 地址,出错
EXAM_MEM:                    LCALL BCD_PBCD2 ;外部存储器检查
                                MOV A,7AH
                                JNB ACC.4,EXAM_NEXT
                                AJMP EXAM3
EXAM_NEXT:                    INC DPTR
                                LCALL PBCD_BCD2 ;下一单元地址送显示
EXAM3:                        CLR A
                                MOVC A,@A+DPTR ;内容送显示
                                MOV R0,#79H
                                LCALL PBCD_BCD
EXAM_GET:                    LCALL KEY_GET ;调键输入子程序
                                JBC ACC.4,EXAM_END
                                MOV 7AH,79H 数字键处理
                                MOV 79H,A
                                AJMP EXAM_GET
EXAM_ERR:                    LCALL ERR_DISP
                                SETB 7CH
EXAM_END:                    RET
EXAM_REG:                    MOV R0,#7BH
                                LCALL BCD_PBCD
                                MOV DPL,A ;映像区低位地址送 DPL
                                MOV DPH,#0E2H ;映像区高位地址送 DPH
                                MOV A,7AH
                                JB ACC.4,EXAM_REG2
EXAM_REG1:                    INC DPTR
                                MOV A,DPL
                                MOV R0,#7BH
                                LCALL PBCD_BCD ;下一单元低位地址送显示
EXAM_REG2:                    MOVX A,@DPTR ;取出内容并送显示
                                MOV R0,#79H
                                LCALL PBCD_BCD
                                AJMP EXAM_GET
;=====
;*      键处理模块   READ      *
;=====
; 功能:检查存储器或片内 RAM 上一单元的内容;然后等待输入,若输入
;      数字键,将低 2 位显示器内容左移一位,键值送入最低位, 然后

```

; 继续等待输入;若输入功能键,跳至主控模块的功能处理部分.
 ; 入口:要求高 4 位显示缓冲区中为存储器或片内 RAM(用其中的低 2 位)
 ; 单元的地址,低 2 位显示缓冲区中为上述单元的内容.
 ; 出口:得到功能键后,到主控模块的功能键处理部分; 不满足入口要
 ; 求时显示错误信息后返回主控模块.

```

READ:          JNB 78H, LAST
                LJMP REC ;上档功能操作,到接收模块

LAST:          CLR 7CH
                LCALL LASTS
                JNB 7CH, LASTC
                LJMP MONITOR

LASTC:         LJMP COMMAND

LASTS:         MOV A, 7AH
                JB ACC.4, LAST_ERR ;本单元内容未显示,出错
                MOV A, 7EH
                JB ACC.4, LAST_REG
                LCALL BCD_PBCD2 ;求出本单元地址
                LCALL DECDPTR ;地址减 1
                LCALL PBCD_BCD2 ;地址送显示
                AJMP EXAM3 ;到显示内容并等待输入

LAST_ERR:      LCALL ERR_DISP
                SETB 7CH
                RET

LAST_REG:      MOV A, 7DH
                JNB ACC.4, LAST_ERR
                MOV A, 7CH
                JB ACC.4, LAST_ERR
                MOV DPH, #0E2H
                MOV R0, #7BH
                LCALL BCD_PBCD
                MOV DPL, A
                LCALL DECDPTR ;地址减 1
                MOV A, DPL
                MOV R0, #7BH
                LCALL PBCD_BCD ;地址低位送显示
                AJMP EXAM_REG2 ;到显示内容并等待输入
  
```

; =====

; * 键处理模块 WRIT *

; =====

; 功能:修改存储器或片内 RAM 单元的内容; 然后显示下一单元地址及
 ; 其中的内容,并等待按键,对按键的处理与 EXAM 模块相同.
 ; 入口:要求高 4 位显示缓冲区中为存储器或片内 RAM 单元的地址, 低 2
 ; 位显示缓冲区中为要存入该地址中的内容.
 ; 出口:同 EXAM 模块.

```

                                ORG 01A4H
WRIT:                          JNB 78H,WRITD
                                LJMP SEND ;上档功能键操作,到发送模块
WRITD:                         CLR 7CH
                                ACALL WRITS
                                JNB 7CH,WRITC
                                LJMP MONITOR
WRITC:                         LJMP COMMAND
WRITS:                         MOV A,7AH
                                JB ACC.4,WRIT_ERR
                                MOV A,7EH
                                JB ACC.4,WRIT_REG
                                LCALL BCD_PBCD2 ;地址送 DPTR
                                MOV R0,#79H
                                LCALL BCD_PBCD ;内容送 A
                                MOVX @DPTR,A
                                AJMP EXAM_NEXT ;到显示下一地址及其中内容
                                    ;并等待输入
WRIT_ERR:                      LCALL ERR_DISP
                                SETB 7CH
                                RET
WRIT_REG:                      MOV A,7DH
                                JNB ACC.4,WRIT_ERR
                                MOV A,7CH
                                JB ACC.4,WRIT_ERR
                                MOV DPH,#0E2H
                                MOV R0,#7BH
                                LCALL BCD_PBCD
                                MOV DPL,A
                                MOV R0,#79H
                                LCALL BCD_PBCD
                                MOVX @DPTR,A
                                AJMP EXAM_REG1 ;到显示下一地址及其中内容
                                    ;并等待输入
;=====
;*      键处理模块      EXEC      *
;=====
; 功能:从指定地址开始全速执行程序.
; 入口:要求高 4 位显示缓冲区中为程序的起始地址.
; 出口:执行该段程序;不满足入口要求时显示错误信息后返回主控.
                                ORG 00AFH
EXEC:                          JNB 78H,EXECD
                                LJMP STEP ;上档功能操作,到单步
EXECD:                         ACALL EXECS

```

```

                                LJMP MONITOR
EXEC_ERR:                      LCALL ERR_DISP
                                RET
EXECS:                          MOV A,7EH
                                JB ACC.4,EXEC_ERR ;起始地址未送,出错
                                MOV R0,#7BH ;求出起始地址
                                LCALL BCD_PBCD
                                MOV 3EH,A
                                LCALL BCD_PBCD
                                MOV 3FH,A
                                LCALL OFF_DISP
                                MOV DPTR,#0E600H ;显示器位口送 DPTR
                                MOV A,#20H
                                MOVX @DPTR,A ;选通最高位
                                MOV DPTR,#0E500H ;显示器数据口送 DPTR
                                MOV A,#0CEH
                                MOVX @DPTR,A ;显示"P"
                                ANL PSW,#00H
                                PUSH 3EH
                                PUSH 3FH
                                RETI ;交出控制权

;=====
; *      键处理模块      DOWN      *
;=====
; 功能;取消尚未执行的命令,强迫功能键进入下档功能.
; 入口;无条件.
; 出口;功能键进入下档功能操作,返回主控模块.
                                ORG 00A0H
DOWN:                            ACALL DOWND
                                LJMP MONITOR
DOWND:                          LCALL OFF_DISP
                                MOV 7EH,#10H ;最高位送"P"
                                MOV 2FH,#00H ;清除监控标志单元
                                RET

;=====
; *      键处理模块      UP      *
;=====
; 功能;强迫功能键进入上档功能.
; 入口;无条件.
; 出口;功能键进入上档功能操作,返回主控模块.
                                ORG 01EEH
UP:                              ACALL UPS
                                LJMP MONITOR
UPS:                            SETB 78H ;设置上档功能操作.

```

```

                                LCALL OFF_DISP
                                MOV 7EH,#1BH
                                RET
;=====
;*      键处理模块      MOV      *
;=====
; 功能:将指定的数据块拷贝到指定的存储地址中.
; 入口:3AH 到 3BH 中为数据块的源首地址,3CH 到 3DH 中为数据块的源末
;       地址,高 4 位显示器中为目标首地址.
; 出口:显示目标首地址及其中的内容后返回主控模块;不满足入口要
;       求时显示错误信息后返回主控.
                                ORG 0242H
MOV:                                JNB 78H,MOVD
                                LJMP EEPROM;上档功能操作,到 EEPROM 写入
MOVD:                                CLR 7CH
                                ACALL MOVS
                                JNB 7CH,MOVC
                                LCALL ERR_DISP
MOVC:                                LJMP MONITOR
MOVS:                                ACALL CHECK_ADDR;检查源地址合法性
                                JZ MOV_ERR
                                JNC MOV_ERR
                                MOV A,7EH
                                JB ACC.4,MOV_ERR;目标首地址未送,出错
                                MOV R0,#7BH
                                LCALL BCD_PBCD
                                MOV 3EH,A;目标首地址低位送 3EH
                                LCALL BCD_PBCD
                                MOV 3FH,A;目标首地址高位送 3FH
                                ACALL MOV_B;移动
                                MOV R1,#3EH
                                ACALL MOV_DISP;显示目标首址及其中内容
                                RET
MOV_ERR:                                SETB 7CH
                                RET
CHECK_ADDR:                                JNB 79H,C_A_ERR
                                JNB 7AH,C_A_ERR
                                MOV R2,3BH
                                MOV R3,3AH
                                MOV R4,3DH
                                MOV R5,3CH
                                ACALL C_A1;比较源首址,源末址大小
                                RET
C_A_ERR:                                CLR A

```



```

C_A1:      RET
           MOV A,R2
           MOV 7FH,R4
           CJNE A,7FH,C_A1_OK
           MOV A,R3
           MOV 7FH,R5
           CJNE A,7FH,C_A1_OK
           CLR A
           RET
C_A1_OK:   MOV A,#0FFH
           RET
MOV_B:     MOV R4,3FH
           MOV R5,3EH
           ACALL C_A1 ;比较源首址与目标首址大小
           JNZ MOV_R ;不等,移动
           RET ;相等,返回
MOV_R:     JC MOV_C ;源首址<目标首址,反向移动
           MOV A,R5
           JNZ MOV_R1
           DEC R4
MOV_R1:    DEC R5
           MOV DPL,R5
           MOV DPH,R4
           MOV R4,3DH
           MOV R5,3CH
           MOV A,R3
           JNZ MOV_R2
           DEC R2
MOV_R2:    DEC R3
NEXT_BYTE: LCALL C_A1 ;比较源地址指针是否移到源末址
           JNZ MOV_R3 ;未到,移动下一字节
           RET ;已到,返回
MOV_R3:    INC DPTR ;移动一个字节
           PUSH DPL
           PUSH DPH
           ACALL GET_BYTE ;源地址指针加 1
           CLR A
           MOVC A,@A+DPTR ;取出源数据
           POP DPH
           POP DPL
           MOVX @DPTR,A ;写入目标地址中
           AJMP NEXT_BYTE
MOV_C:     MOV A,3CH ;反向移动
           CLR C ;计算数据块长度

```

```

SUBB A,R3
MOV R1,A
MOV A,3DH
SUBB A,R2
MOV R0,A
MOV A,R5
SETB C ;计算目标末地址
ADDC A,R1
MOV R7,A
MOV A,R4
ADDC A,R0
MOV R6,A
MOV R4,3DH
MOV R5,3CH
INC R5
MOV A,R5
JNZ MOV_C1
INC R4
MOV_C1:      ACALL C_A1 ;比较源地址指针是否移到源首址
JNZ MOV_C2 ;未到,移动下一字节
RET ;已到,返回
MOV_C2:      MOV A,R7 ;移动一个字节
JNZ MOV_C3
DEC R6
MOV_C3:      DEC R7
MOV A,R5
JNZ MOV_C4
DEC R4
MOV_C4:      DEC R5
MOV DPL,R5
MOV DPH,R4
CLR A
MOVC A,@A+DPTR
MOV DPH,R6
MOV DPL,R7
MOVX @DPTR,A
AJMP MOV_C1
MOV_DISP:    MOV DPL,@R1
INC R1
MOV DPH,@R1
LCALL PBCD_BCD2
MOVX A,@DPTR
MOV R0,#79H
LCALL PBCD_BCD

```

```

                                RET
GET_BYTE:                      INC R3
                                MOV A,R3
                                JNZ G_B1
                                INC R2
G_B1:                          MOV DPH,R2
                                MOV DPL,A
                                RET

```

```

;=====
;*      键处理模块      F      *
;=====

```

```

; 功能:将数据块操作中源数据块的首地址和末地址依次存入 3AH 到
;      3BH 和 3CH 到 3DH 中.
; 入口:要求地址已送入高 4 位显示器中.
; 出口:若源首址和源末址已存入 3AH 到 3DH 中,直接返回主控模块, 否
;      则将地址存入后返回主控模块;不满足入口要求则显示错误信
;      息后返回主控模块.

```

```

                                ORG 01FCH
F:                              MOV A,7EH
                                JB ACC.4,F_ERR
                                JNB 79H,F_10 ;源首址未存,到存源首址
                                JNB 7AH,F_20 ;源末址未存,到存源末址
                                SJMP F_END
F_ERR:                          LCALL ERR_DISP
F_END:                          LJMP MONITOR
F_10:                           ACALL F1
                                SJMP F_END
F_20:                           LCALL F2
                                SJMP F_END
F1:                             MOV R0,#7BH
                                LCALL BCD_PBCD
                                MOV 3AH,A ;源首址低位送 3AH
                                LCALL BCD_PBCD
                                MOV 3BH,A ;源首址高位送 3BH
                                SETB 79H ;设置源首址已存标志
                                LCALL OFF_DISP
                                MOV 7EH,#11H
                                RET
F2:                             MOV R0,#7BH
                                LCALL BCD_PBCD
                                MOV 3CH,A ;源末址低位送 3CH
                                LCALL BCD_PBCD

```

```

MOV 3DH,A ;源末址高位送 3DH
SETB 7AH ;设置源末址已存标志
LCALL OFF_DISP
MOV 7EH,#12H
RET

```

```

;=====
;*      单步模块      STEP      *
;=====

```

```

; 功能:单步跟踪执行程序.
; 入口:高 4 位显示器中为程序的起始地址.
; 出口:执行此地址处的一条指令后显示累加器 A 的值及下一条指令的
;      地址.
; 说明:单步调试时不能修改监控标志单元 2FH.

```

```

                                ORG 0470H
STEP:                            JNB 7BH,STEP0
                                AJMP STEP1
STEP0:                          MOV R0,#7BH ;第一次进入单步
                                LCALL BCD_PBCD
                                PUSH ACC ;起始地址低位压入堆栈
                                LCALL BCD_PBCD
                                PUSH ACC ;起始地址高位压入堆栈
                                SETB 7BH ;设置已进入单步标志
                                AJMP DRT
STEP1:                          MOV R0,#7DH ;非第一次进入单步,先恢复片内
                                ;RAM 单元
                                MOV R1,#02H
                                MOV DPTR,#0E202H
STEP2:                          MOVX A,@DPTR
                                MOV @R1,A
                                INC DPTR
                                INC R1
                                DJNZ R0,STEP2
                                MOV DPTR,#0E200H
                                MOVX A,@DPTR
                                MOV R0,A
                                INC DPTR
                                MOVX A,@DPTR
                                MOV R1,A
                                MOV DPTR,#0E281H
                                MOVX A,@DPTR
                                MOV 81H,A
                                MOV DPTR,#0E288H

```

MOVX A,@DPTR
MOV 88H,A
INC DPTR
MOVX A,@DPTR
MOV 89H,A
INC DPTR
MOVX A,@DPTR
MOV 8AH,A
INC DPTR
MOVX A,@DPTR
MOV 8BH,A
INC DPTR
MOVX A,@DPTR
MOV 8CH,A
INC DPTR
MOVX A,@DPTR
MOV 8DH,A
MOV DPTR,#0E2A8H
MOVX A,@DPTR
MOV 0A8H,A
MOV DPTR,#0E2B8H
MOVX A,@DPTR
MOV 0B8H,A
MOV DPTR,#0E2C8H
MOVX A,@DPTR
MOV 0C8H,A
INC DPTR
INC DPTR
MOVX A,@DPTR
MOV 0CAH,A
INC DPTR
MOVX A,@DPTR
MOV 0CBH,A
INC DPTR
MOVX A,@DPTR
MOV 0CCH,A
INC DPTR
MOVX A,@DPTR
MOV 0CDH,A
MOV DPTR,#0E2D0H
MOVX A,@DPTR
MOV 0D0H,A
MOV DPTR,#0E2F0H
MOVX A,@DPTR

	MOV 0F0H,A
	MOV DPTR,#0E282H
	MOVX A,@DPTR
	PUSH ACC
	INC DPTR
	MOVX A,@DPTR
	PUSH ACC
	MOV DPTR,#0E2E0H
	MOVX A,@DPTR
	POP DPH
	POP DPL
DRT:	SETB EA ;允许中断
	CLR IT0
	SETB PX0
	SETB EX0
	RETI ;执行地址在栈顶的指令
PUSH_REG:	PUSH DPH ;单步中断,先保护片内 RAM 单元
	PUSH DPL
	PUSH ACC
	MOV DPTR,#0E200H
	MOV A,R0
	MOVX @DPTR,A
	INC DPTR
	MOV A,R1
	MOVX @DPTR,A
	MOV R0,#02H
	MOV R1,#7DH
	MOV DPTR,#0E202H
P_R0:	MOV A,@R0
	MOVX @DPTR,A
	INC DPTR
	INC R0
	DJNZ R1,P_R0
	MOV DPTR,#0E2E0H
	POP ACC
	MOVX @DPTR,A
	POP DPL
	POP DPH
	MOV R0,DPL
	MOV R1,DPH
	MOV DPTR,#0E282H
	MOV A,R0
	MOVX @DPTR,A
	INC DPTR

```
MOV A,R1
MOVX @DPTR,A
MOV A,80H
MOV DPTR,#0E280H
MOVX @DPTR,A
MOV A,81H
INC DPTR
MOVX @DPTR,A
INC DPTR
MOV DPTR,#0E287H
MOV A,87H
MOVX @DPTR,A
INC DPTR
MOV A,88H
MOVX @DPTR,A
INC DPTR
MOV A,89H
MOVX @DPTR,A
INC DPTR
MOV A,8AH
MOVX @DPTR,A
INC DPTR
MOV A,8BH
MOVX @DPTR,A
INC DPTR
MOV A,8CH
MOVX @DPTR,A
INC DPTR
MOV A,8DH
MOVX @DPTR,A
INC DPTR
INC DPTR
INC DPTR
MOV A,90H
MOVX @DPTR,A
MOV DPTR,#0E298H
MOV A,98H
MOVX @DPTR,A
INC DPTR
MOV A,99H
MOVX @DPTR,A
MOV DPTR,#0E2A0H
MOV A,0A0H
MOVX @DPTR,A
```

```

MOV DPTR,#0E2A8H
MOV A,0A8H
MOVX @DPTR,A
MOV DPTR,#0E2B0H
MOV A,0B0H
MOVX @DPTR,A
MOV DPTR,#0E2B8H
MOV A,0B8H
MOVX @DPTR,A
MOV DPTR,#0E2C8H
MOV A,0C8H
MOVX @DPTR,A
INC DPTR
INC DPTR
MOV A,0CAH
MOVX @DPTR,A
INC DPTR
MOV A,0CBH
MOVX @DPTR,A
INC DPTR
MOV A,0CCH
MOVX @DPTR,A
INC DPTR
MOV A,0CDH
MOVX @DPTR,A
MOV DPTR,#0E2D0H
MOV A,0D0H
MOVX @DPTR,A
MOV DPTR,#0E2F0H
MOV A,0F0H
MOVX @DPTR,A
JNB 7EH,STEPDISP
LJMP 1200H ;CRT 监控的单步中断，长跳到 CRT
           ;监控相应部分
STEPDISP: POP DPH ;取出下条指令地址
           POP DPL
           PUSH DPL ;保存下条指令地址
           PUSH DPH
           LCALL PBCD_BCD2 ;下条指令地址送显示
           MOV DPTR,#0E2E0H ;A 映像单元的值送 A
           MOVX A,@DPTR
           MOV R0,#79H ;A 的值送显示
           LCALL PBCD_BCD
           LJMP K_P1

```


	ORG 0616H
B1200:	MOV TMOD,#20H MOV TH1,#222 MOV TL1,#222 SETB TR1 AJMP HBEG
B2400:	MOV TMOD,#20H MOV TH1,#239 MOV TL1,#239 SETB TR1 AJMP HBEG
B4800:	MOV TMOD,#20H MOV TH1,#247 MOV TL1,#247 SETB TR1 AJMP HBEG
HBEG:	NOP ;接收浮动地址 ACALL TB_0 MOV R5,A ACALL TB_0 MOV R4,A
H_0:	JBC RI,H_1 ;接收行首标志字符 AJMP H_0
H_1:	MOV A,SBUF ACALL PARITY ;校验 CJNE A,#3AH,REC_ERR
B_0:	ACALL TB_0 ;接收本行有效字节数 MOV R7,A ACALL TB_0 ;接收本行起始地址 MOV DPH,A ACALL TB_0 MOV DPL,A CLR C ;求出本行绝对地址 ADD A,R4 MOV DPL,A MOV A,DPH ADDC A,R5 MOV DPH,A ACALL TB_0 ;丢弃地址后的"00"
D_0:	ACALL TB_0 ;接收一字节机器码 MOVX @DPTR,A INC DPTR DJNZ R7,D_0 ;本行未接收完,继续接收下一字

```

                                ;节
                                ACALL TB_0 ;丢弃行尾校验和
                                ACALL TB_0 ;丢弃回车,换行
L_0:                            JBC RI,L_1
                                AJMP L_0
REC_ERR:                        LCALL ERR_DISP
                                LJMP MONITOR
L_1:                            MOV A,SBUF
                                ACALL PARITY
                                CJNE A,#00H,B_0 ;不是结束符,继续接收下一
                                ;行
                                MOV A,#00H ;接收完毕,返回初始化模块
                                PUSH ACC
                                PUSH ACC
                                RETI
TB_0:                           JBC RI,TB_1 ;接收 2 个 ASC 字符,将其转换成压
                                ;缩 BCD 码后返回
                                AJMP TB_0
TB_1:                           MOV A,SBUF
                                ACALL PARITY
                                MOV R6,A
TB_2:                           JBC RI,TB_3
                                AJMP TB_2
TB_3:                           MOV A,SBUF
                                ACALL PARITY
                                ACALL A_DB
                                RET
A_DB:                           ACALL A_B ;将 2 个 ASC 字符转换为压缩 BCD 码
                                XCH A,R6
                                ACALL A_B
                                SWAP A
                                ORL A,R6
                                RET
PARITY:                         MOV C,RB8 ;偶校验
                                ANL C,PSW.0
                                JC COBK
                                MOV C,RB8
                                ORL C,PSW.0
                                JNC COBK
                                AJMP REC_ERR
COBK:                           RET
A_B:                            CLR C ;ASC 码转换为 BCD 码
                                CJNE A,#39H,A_CON
                                AJMP A_NUM

```

```

A_CON:          JNC A_AL
A_NUM:          CLR C
                SUBB A,#30H
                AJMP A_BRET
A_AL:           CLR C
                SUBB A,#55
A_BRET:         RET

```

```

;=====
;*      接收模块      REC      *
;=====

```

```

; 功能:与高位机的发送模块配合,将用户的目标程序装入存储器中.
; 入口:无条件.
; 出口:接收完毕后返回初始化模块, 若接收中发现错误,显示错误信
;      息后返回主控模块.

```

```

                ORG 05E0H
REC:            MOV TMOD,#20H ;初始化串行口
                MOV TL1,#0BBH
                MOV TH1,#0BBH
                SETB TR1
                CLR ET1
                MOV PCON,#80H
                MOV SCON,#0D0H
HB0:            JBC RI,HB1
                AJMP HB0
HB1:            MOV A,SBUF ;接收波特率选择因子
                DEC A
                CLR C
                RLC A
                MOV DPTR,#BAUD
                JMP @A+DPTR ;根据选择因子,重新设置波特率
BAUD:           AJMP B600
                AJMP B1200
                AJMP B2400
                AJMP B4800
B600:          MOV TMOD,#20H
                MOV TH1,#0BBH
                MOV TL1,#0BBH
                SETB TR1
                AJMP HBEG

```

```

;=====
;*      发送模块      SEND      *
;=====

```

```

; 功能:与高位机的接收模块配合,将存储器中的一段数据或程序发送
;      给高位机.

```

；入口:3AH 到 3BH 中为数据块的起始地址,3CH 到 3DH 中为数据块的结束
；地址,高 4 位显示器中为数据块的重定位地址.
；出口:发送完毕后返回初始化模块;不满足入口要求时显示错误信息
；后返回主控模块.

```
ORG 06E0H
SEND:    NOP
        MOV A,7EH
        JB ACC.4,SEND_ERR ;未建立目标地址,出错
        MOV R0,#7BH ;保存目标地址
        LCALL BCD_PBCD
        MOV 3EH,A
        LCALL BCD_PBCD
        MOV 3FH,A
        MOV TMOD,#20H ;初始化串行口
        MOV TH1,#0BH
        MOV TL1,#0BBH
        SETB TR1
        CLR ET1
        MOV SCON,#0D0H
        MOV PCON,#80H
        MOV R2,3AH ;计算目标末地址
        MOV DPL,R2
        MOV R1,3BH
        MOV DPH,R1
        MOV R7,3CH
        MOV R6,3DH
        MOV A,R7
        CLR C
        SUBB A,R2
        MOV R7,A
        MOV A,R6
        SUBB A,R1
        MOV R6,A
        CLR C
        MOV A,3EH
        ADDC A,R7
        MOV 38H,A
        MOV A,3FH
        ADDC A,R6
        MOV 39H,A
        MOV R2,#02H
        MOV R1,#3FH
SEND_ADDRB:    MOV A,@R1 ;发送目标起始地址
        MOV C,P
```

```

MOV TB8,C
MOV SBUF,A
S_AB0:    JBC TI,S_AB1
          AJMP S_AB0
SEND_ERR:    LCALL ERR_DISP
          LJMP MONITOR
S_AB1:    DEC R1
          DJNZ R2,SEND_ADDRB
          MOV R2,#2
          MOV R1,#39H ;发送目标末地址
SEND_ADDRE:    MOV A,@R1
          MOV C,P
          MOV TB8,C
          MOV SBUF,A
S_AE0:    JBC TI,S_AE1
          AJMP S_AE0
S_AE1:    DEC R1
          DJNZ R2,SEND_ADDRE
SEND_DATA:    MOV A,#00H ;发送数据
          MOVC A,@A+DPTR
          MOV C,P
          MOV TB8,C
          MOV SBUF,A
S_D0:    JBC TI,S_D1
          AJMP S_D0
S_D1:    DJNZ R7,S_DNEXT
          DJNZ R6,S_DNEXT
          AJMP SEND_END
S_DNEXT:    INC DPTR
          AJMP SEND_DATA
SEND_END:    MOV A,#00H ;发送完毕,返回初始化模块
          PUSH ACC
          PUSH ACC
          RETI
;=====
; *      写入模块      EEPROM      *
;=====
; 功能:将存储器中的一段程序写入 EEPROM 中.
; 入口:3AH 到 3BH 中为程序段的起始地址,3CH 到 3DH 中为程序段的结束
;       地址,高 4 位显示器中为写到 EEPROM 中的起始地址.
; 出口:写完后返回初始化模块;不满足入口要求时显示错误信息后返
;       回主控模块.
          ORG 0770H
EEPROM:    NOP

```

```

MOV A,7EH
JB ACC.4,EEP_ERR ;目标首址未送,出错
MOV R0,#7BH
LCALL BCD_PBCD
MOV 3EH,A ;目标首址低位送 3EH
LCALL BCD_PBCD
MOV 3FH,A ;目标首址高位送 3FH
MOV R4,3BH
MOV R5,3AH
MOV R6,3DH
MOV R7,3CH
EEPROM1:    CLR C ;判断源地址指针是否移到源末址
MOV A,R6
SUBB A,R4
CJNE A,#00H,EEPROM2 ;未到,继续写入一个字
                ;节

CLR C
MOV A,R7
MOV A,R5
CJNE A,#00H,EEPROM2
AJMP EEPROM3 ;已到,返回初始化模块
EEP_ERR:    LCALL ERR_DISP
LJMP MONITOR
EEPROM2:    MOV DPH,R4
MOV DPL,R5
ACALL EE_BYTE ;写入一个字节
MOV R4,DPH
MOV R5,DPL
AJMP EEPROM1
EEPROM3:    MOV A,#00H ;EEPROM 写入完毕, 返回初始化模
                ;块

PUSH ACC
PUSH ACC
RETI
EE_BYTE:    PUSH PSW ;写入一个字节并校验
SETB RS0
SETB RS1
MOV A,#00H
MOVC A,@A+DPTR
PUSH DPL
PUSH DPH
MOV DPH,3FH
MOV DPL,3EH
MOVX @DPTR,A ;写入

```

```

                                ACALL EEPROM_DELAY
                                MOV R2,A
EEPROM4:                        MOVX A,@DPTR
                                XRL A,R2
                                CJNE A,#0,EEPROM4 ;校验
                                INC DPTR
                                MOV 3FH,DPH
                                MOV 3EH,DPL
                                POP DPH
                                POP DPL
                                INC DPTR
                                POP PSW
                                RET
EEPROM_DELAY:                   MOV R7,#02H
EE_D1:                          MOV R6,#0FFH
EE_D2:                          DJNZ R6,$
                                DJNZ R7,EE_D1
                                RET

;=====
;*      固化模块      EPROM      *
;=====
; 功能:将存储器中的一段程序固化到 EPROM 中.
; 入口:同 EEPROM 模块.
; 出口:同 EEPROM 模块.
                                ORG 07F0H
EPROM:                          PUSH ACC
                                PUSH PSW
                                SETB RS0
                                SETB RS1
                                PUSH DPL
                                PUSH DPH
                                MOV DPTR,#0E200H ;8155 命令口地址送 DPTR
                                MOV A,#0FH
                                MOVX @DPTR,A ;设置 A,B,C 口全为输出
                                CLR P3.5
                                SETB P3.4
                                MOV A,7EH
                                JB ACC.4,EP_ERR
                                JNB ACC.4,CONTINUE
EP_ERR:                         LCALL ERR_DISP
                                LJMP MONITOR
CONTINUE:                       MOV R0,#7BH ;计算目标首址
                                LCALL BCD_PBCD

```

```

MOV 3EH,A
LCALL BCD_PBCD
MOV 3FH,A
MOV R0,3BH
MOV R1,3AH
MOV R2,3DH
MOV R3,3CH
MOV R4,3FH
MOV R5,3EH
CLR C ;计算程序段长度
MOV A,R3
SUBB A,R1
MOV R7,A
MOV A,R2
SUBB A,R0
MOV R6,A
MOV DPL,R1
MOV DPH,R0
EPROM1:    MOV A,#00H
            MOVC A,@A+DPTR ;取出一字节
            INC DPTR
            PUSH DPL
            PUSH DPH
            MOV DPTR,#0E201H
            MOVX @DPTR,A ;将其从 A 口输出
            MOV R0,#121
            LCALL PBCD_BCD
            INC DPTR
            MOV A,R5
            MOVX @DPTR,A ;地址低位从 B 口输出
            MOV R0,#123
            LCALL PBCD_BCD
            INC DPTR
            MOV A,R4
            MOVX @DPTR,A ;地址高位从 C 口输出
            MOV R0,#125
            LCALL PBCD_BCD
            MOV A,R5
            INC A
            MOV R5,A
            JNZ EPROM2
            INC R4
EPROM2:    SETB P3.5
            CLR P3.4

```



```

        PUSH PSW
        CLR RS0
        SETB RS1
        LCALL LED_DISPLAY ;调显示延时
        LCALL LED_DISPLAY
        LCALL LED_DISPLAY
        LCALL LED_DISPLAY
        LCALL LED_DISPLAY
        LCALL LED_DISPLAY
        LCALL LED_DISPLAY
        LCALL LED_DISPLAY
        POP PSW
        CLR P3.5
        SETB P3.4
        POP DPH
        POP DPL
        CJNE R6,#0,EPROM3 ;未完,继续固化下一字节
        CJNE R7,#0,EPROM3 ;完毕,返回初始化模块
        MOV A,#0
        PUSH ACC
        PUSH ACC
        RETI
EPROM3:    MOV A,R7
        CLR C
        SUBB A,#01H
        MOV R7,A
        MOV A,R6
        SUBB A,#00H
        MOV R6,A
        LJMP EPROM1
;=====
; * 关闭显示子程序 OFF_DISP *
;=====
; 功能:将全暗的数据信息送到 6 个显示缓冲区.
; 入口:无条件.
; 出口:6 个显示缓冲区中为全暗的数据信息.
        ORG 0340H
OFF_DISP:    MOV A,#16H ;全暗的数据编码送 A
        MOV 7EH,A
        MOV 7DH,A
        MOV 7CH,A
        MOV 7BH,A
        MOV 7AH,A
        MOV 79H,A

```

```

                                RET
;=====
; *  错误信息显示子程序   ERR_DISP   *
;=====
; 功能:送错误提示"ErrorE"到显示缓冲区.
; 入口:无条件.
; 出口:显示缓冲区中为上述错误提示信息.
                                ORG 034FH
ERR_DISP:                      MOV 7EH,#0EH;"E"的编码
                                MOV 7DH,#18H;"r"的编码
                                MOV 7CH,#18H
                                MOV 7BH,#19H;"o"的编码
                                MOV 7AH,#18H
                                MOV 79H,#0EH
                                RET
;=====
; *  地址减 1 子程序   DECDPTR   *
;=====
; 功能:将存放在 DPTR 中的地址减 1.
; 入口:DPTR 中为原地址.
; 出口:原地址减 1 后仍存放在 DPTR 中返回.
                                ORG 0362H
DECDPTR:                        PUSH ACC
                                MOV A,DPL
                                JNZ DECDPL
                                DEC DPH
DECDPL:                         DEC DPL
                                POP ACC
                                RET
;=====
; * BCD 到压缩 BCD 变换子程序 1   BCD_PBCD   *
;=====
; 功能:将 2 位 BCD 码变成压缩 BCD 码.
; 入口:R0 中含有低位 BCD 码的地址.
; 出口:A 中为得到的压缩 BCD 码.
                                ORG 036FH
BCD_PBCD:                       MOV A,@R0;取出低位 BCD 码
                                INC R0
                                SWAP A
                                ORL A,@R0;取出高位 BCD 码
                                SWAP A
                                INC R0
                                RET
;=====

```

; * BCD 到压缩 BCD 变换子程序 2 BCD_PBCD2 *

;

; 功能:将 4 位 BCD 码变成压缩 BCD 码.

; 入口:7BH 到 7EH 中为从低到高的 4 位 BCD 码.

; 出口:DPTR 中为得到的压缩 BCD 码.

ORG 0376H

BCD_PBCD2: MOV R0,#7BH

ACALL BCD_PBCD

MOV DPL,A ;地址低位送 DPL

ACALL BCD_PBCD

MOV DPH,A ;地址高位送 DPH

RET

;

; * 压缩 BCD 到 BCD 变换子程序 1 PBCD_BCD *

;

; 功能:将 1 字节压缩 BCD 码变成 2 位 BCD 码.

; 入口:A 为需变换的压缩 BCD 码,R0 中为变换后低位 BCD 码存放的地址.

; 出口:R0 指定的地址中存放着 BCD 码.

ORG 0381H

PBCD_BCD: MOV @R0,A ;取出压缩 BCD 码

ANL A,#0FH ;屏蔽高半字节

XCH A,@R0 ;将@R0 换为 BCD 码低字节,A 得到压
;缩 BCD 码

INC R0

SWAP A

ANL A,#0FH

MOV @R0,A ;将高位 BCD 码送入@R0 中

INC R0

RET

;

; * 压缩 BCD 到 BCD 变换子程序 2 PBCD_BCD2 *

;

; 功能:将 2 字节压缩 BCD 码变成 BCD 码.

; 入口:压缩 BCD 码已存放到 DPTR 中.

; 出口:得到的 BCD 码从低到高存放在 7BH 到 7EH 中.

ORG 038CH

PBCD_BCD2: MOV A,DPL

MOV R0,#7BH

ACALL PBCD_BCD ;地址低位送显示

MOV A,DPH

ACALL PBCD_BCD ;地址高位送显示

RET

```

;=====
;* 显示子程序 LED_DISPLAY *
;=====
; 功能:将显示缓冲区的数据送到数码管上显示.
; 入口:显示缓冲区中已存入要求显示的数据.
; 出口:数码管上显示出要求显示的数据.

                ORG 03A0H
LED_DISPLAY:    MOV R0,#79H ;显缓地址送 R0
                MOV R3,#01H ;显示器选通值送 R3
                MOV A,R3
LED_DISPLAY1:   MOV DPTR,#0E600H
                MOVX @DPTR,A ;选通一位显示器
                MOV A,@R0 ;计算显示数据的显示编码
                ADD A,#1BH
                MOVC A,@A+PC
                MOV DPTR,#0E500H
                MOVX @DPTR,A
                LCALL DELAY
                INC R0
                MOV A,R3
                JB ACC.5,LED_DISPLAY2 ;已到最高位显示器,
                                     ;返回
                RL A ;否则,选通下一位显示器
                MOV R3,A
                LJMP LED_DISPLAY1
LED_DISPLAY2:   RET
DELAY:          MOV R7,#02H
DELAY1:         MOV R6,#0FFH
DELAY2:         DJNZ R6,DELAY2
                DJNZ R7,DELAY1
                RET
                DB 0C0H,0F9H,0A4H,0B0H,99H,92H
                DB 82H,0F8H,80H,90H,88H,83H
                DB 0C6H,0A1H,86H,8EH,8CH,0DEH
                DB 0F3H,91H,0BFH,7FH,0FFH,00H
                DB 0AFH,0A3H,00H
                DB 0CH,0C1H
;=====
;* 键输入子程序 KEY_GET *
;=====
; 功能:在维持显示的同时等待用户按键; 有键按下后,取得最先被按
;       下的键的键号;在所有按键均被释放后返回.
; 入口:无条件.
; 出口:A 中为最先被按下的键的键号.

```

```

                                ORG 0400H
KEY_GET:                        LCALL KEY_SCAN;调键扫描子程序
                                JNZ K_G1 ;有键按下转下一步处理
                                LCALL LED_DISPLAY ;无键按下调显示子程序
                                AJMP KEY_GET ;继续等待
K_G1:                          LCALL LED_DISPLAY ;两次调用显示子程序
                                LCALL LED_DISPLAY ;去抖动
                                LCALL KEY_SCAN
                                JNZ K_G2 ;仍有键按下转键识别
                                LCALL LED_DISPLAY
                                AJMP KEY_GET
K_G2:                          MOV R2,#0FEH ;列选通初始化,选通第 1 列
                                MOV R4,#00H ;列初始值为 00H
K_G3:                          MOV DPTR,#0E600H
                                MOV A,R2
                                MOVX @DPTR,A ;选通一列
                                MOV DPTR,#0E700H
                                MOVX A,@DPTR ;读入键盘状态
                                JB ACC.0,LINE1
                                MOV A,#00H ;第 1 行首键号 00H
                                ;到 A
                                AJMP K_G_END
LINE1:                          JB ACC.1,LINE2
                                MOV A,#08H ;第 2 行首键号为 08H
                                AJMP K_G_END
LINE2:                          JB ACC.2,NEXT_COL ;本列无键按下,检查下列
                                MOV A,#10H ;第 3 行首键号为 10H
K_G_END:                        ADD A,R4 ;首键号+列值=键号
                                PUSH ACC
KEY_FREE:                       LCALL LED_DISPLAY ;等待释放全部按键
                                LCALL KEY_SCAN
                                JNZ KEY_FREE
                                POP ACC
                                RET
NEXT_COL:                       INC R4 ;列值加 1
                                MOV A,R2
                                JNB ACC.7,KEY_NEXT ;刚才为最后一列,重新
                                ;等待输入
                                RLA ;列选通左移一位,即选通下一列
                                MOV R2,A
                                AJMP K_G3
KEY_NEXT:                       AJMP KEY_GET
;=====
; * 键扫描子程序 KEY_SCAN *

```



```
POP ACC
RET
ADDD1:    MOV 7EH,#0AH
          MOV 7DH,#0DH
          MOV 7CH,#14H
          MOV 7BH,39H ;通道号送 7BH
          MOV R0,#79H
          MOV A,38H ;结果送低 2 位显示器
          LCALL PBCD_BCD
          RET
AD_DELAY:    MOV R0,#0FH
          MOV R1,#10H
AD_D1:    DJNZ R1,AD_D1
          MOV R1,#10H
          DJNZ R0,AD_D1
          RET
          END
```