多功能自行车测速仪

目录

一、产品设计背景……2
二、产品功能简介……2
三、系统硬件设计……3
四、MCU 软件设计……8
五、上位机软件 bike V1.0 设计……8
六、产品实物及测试……10
6．1<测速模式>……13
6．2<数据传输>……14
6．3<其它功能>……15
6．4<退出系统>……16
七、结语……16
附录 1 电路图……17
附录 2 源程序……18


Disc Brakes
Tire
Quick Release
Spoke
Rim
Hubs
Front Brake
Front Fork
Valve
Front fork end
Nipple
Grip/Tape
Bar End
Brake Lever
Brake Cable Housing
Fork Crown
Brake Cable
Handlebar
Derailleur Lever
Stem
Head Parts
Head Tube
Decals
Down Tube
Top Tube
Rear Suspension
Front Derailleur
Chainwheel & Cranks
Pedal
Seat Tube
Chain
Seat Post
Seat Post Clamp
Saddle
Rear Brake
Chain Stay
Rear Derailleur
Seat Stay
Freewheel
Rear Ends/Dropouts
Derailleur Cable
Rim

多功能自行车测速仪使用说明书

一、产品设计背景

随着人们生活水平的逐渐提高，人们对于生活质量的要求也日益增加，尤其是对健身的要求。自行车在中国普遍作为代步工具。而在国外，自行车却是一项十分受欢迎的健身运动。因为它无污染，价位低廉，老少皆宜。而且在运动过程中可以充分享受到大自然，对于忙碌的现代人来说，无疑是一种较好的放松方法。在中国这种情况也在慢慢发生变化。因此爱好自行车运动的人十分学要一款能测速的装置，以知道自己的运动情况。并根据外界条件，如温度，风速等进行适当的调节，已达到最佳运动的效果。

而对于自行车运动员来说，最为关心的莫过于一段时间内的训练效果。因为教练要根据一段时间内运动员的训练效果进行评估，从而进行适当的调整已使运动员达到最佳的状态。因此需要一种装置进行对训练中各种参数的测定记录。本作品就是针对此而设计的。

二、产品功能简介

- 1. 对自行车进行实时速度的测量。显示出速度值。
- 2. 能针对不同的车型进行选择。从而采用不同的模块进行测量。
- 3. 能测量出当前环境的温度，以供使用者决定是否适宜进行运动。
- 4. 显示当前日期时间，可以任意设定当前工作时间。
- 5. 显示行车里程，运动时间。

6. 可以自行设定采样频率

7. 记录一段时间内的定时采样速度，存入制定单元。通过与 PC 机进行通讯，将数据传送到 PC 机中用如见进行处理，分析。得出运动或训练的情况。

8. 配套软件 bike v1.0 可以将本次运动的速度绘制成速度曲线，以供参考。并可以将数据转存入数据库保存以备日后查询使用

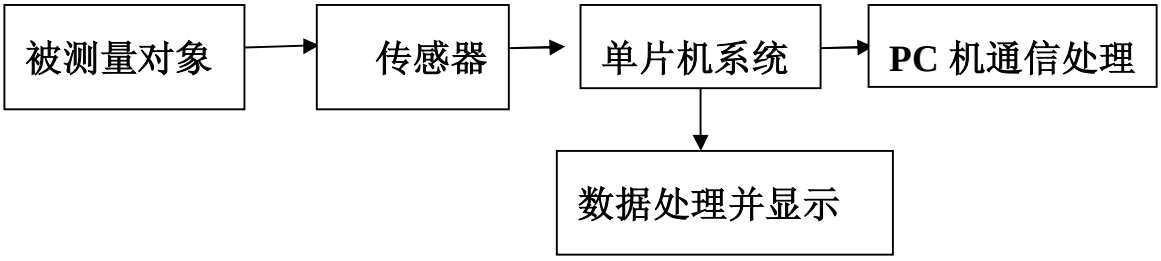
9. 配套软件 bike v1.0 充分考虑到广大自行车爱好者对于自行车运动的热衷，因此加入了对自行车运动的介绍，当今流行车型的简介以及进行自行车运动的注意事项和自行车旅行的相关知识。

并会逐渐对该软件加以升级，使其功能更加完善，以满足广大使用者的需求。

10. 可以进入系统休眠方式以节省电能，并随时激活唤醒系统重新进行工作。可以调节液晶对比度，可以打开背景灯显示。

三、系统硬件设计

系统框图



通过传感器对外部物理量进行测量，再将物理信号转换为电信号，

输入单片机，单片机对所输入的电信号进行处理，最后输出显示，并可以通过与上位机通讯将数据采集到电脑中。

本设计中用到的主要部件包括单片机 **AT89C52**、**DS12887** 时钟芯片、**DS18B20** 温度传感器、欧姆龙公司的 **EE-SX671** 型光电传感器、MAX232 通信芯片以及液晶显示器。

1、 时钟芯片 DS12887

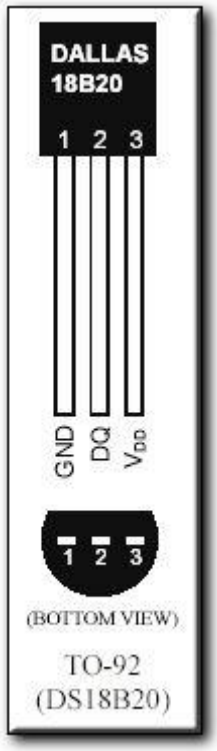
DS12887 是美国达拉斯半导体公司最新推出的时钟芯片，采用 CMOS 技术制成，把时钟芯片所需的晶振和外部锂电池相关电路集于芯片内部。采用 DS12887 芯片设计的时钟电路勿需任何外围电路并具有良好的微机接口。DS12887 芯片具有微功耗、外围接口简单、精度高、工作稳定可靠等优点，可广泛用于各种需要较高精度的实时时钟场合中。

这里我们将 DS12887 的数据线与单片机的 P0 口相连，将其片选线与 P2.0 相连，这样便可通过 R0 或 R1 寄存器实现时钟芯片与单片机的数据传输。注意在 DS12887 的 RESET 端应连接上电复位电路，因为只有当该脚保持低电平时间大于 200ms，DS12887 才能有效工作。DS12887 的 IRQ 端脚为其中断信号输出口，低电平有效，可作微处理的中断输入。没有中断条件满足时，IRQ 处于高阻态。IRQ 线是漏极开路输入，要求外接上接电阻。时钟芯片的有多种中断处理方式，例如周期中断，闹钟中断，更新中断等，这里我们利用它的更新中断，即时间每过 1 秒中，DS12887 的更新中断便会从 IRQ 端输出，而 IRQ 引脚则与单片机的 INT1 中断相连，这样每当 DS12887 发出时间上的

中断请求单片机便可从它的 INT1 中断得知，随之立刻进入中断子程序，在这个中断程序中对光电传感器所发出的脉冲信号进行计数处理，这样就可以得到速度以及里程等所要测量的量。

由于在测速系统中时间的测定非常重要，因此 DS12887 可以说是本设计的核心部件，它的使用好坏直接影响着最终测量的精确度，所以在调试时务必细心，尽量在熟悉 DS12887 的各种功能后在开始编程。

2、 温度传感器 DS1820



DSl820 数字温度计提供 9 位(二进制)温度读数指示器件的温度信息经过单线接口送入 DSl820 或从 DSl820 送出因此从主机 CPU 到 DSl820 仅需一条线(和地线)DSl820 的电源可以由数据线本身提供而不需要外部电源因为每一个 DSl820 在出厂时已经给定了唯一的序号因此任意多个 DSl820 可以存放在同一条单线总线上这允许在许多不同的地方放置温度敏感器件 DSl820 的测量范围从-55 到+125 增量值为 0.5 可在 l s(典型值)内把温度变换成数字。

由于 DS1820 采用单总线结构因此外围电路非常简单，通过一上拉电阻 R5 即可与单片机相连。这里我们将 DS1820 的数据引脚 DQ 与单片机的 T0 口（即 P3.4）相连，通过这条数据线接收温度测量值。

虽然 DS1820 的外围电路十分简单，但是凡事哪能完美，可以

说这是以繁杂的编程换来的。DS1820 以其严格而繁难的时序要求著称。因为它只有一根数据线，即数据与命令字都要在同一条线上传输，所以 DS1820 制定了严格的时序，大家在使用时务必严格按照说明书上的要求对其进行操作，否则时序要是不正确，DS1820 就罢工。笔者在调试的时候大部分时间都花在了调试 DS1820 上。通过切身的体会，我觉的最好变一到两个通用的延时子程序，在 DS1820 的编程中就可以直接调用延时程序，以保证时序的准确。

3、EE-SX671 型光电传感器

EE-SX671 型光电传感器是欧姆龙公司所生产的光电开关型传感器。其四个引脚中我们只需用其中的三个：电源端，接地端以及信号输出端。在车子行驶过程中，车轮带动码盘旋转，由于码盘上刻有等分的孔，在连续的透光与挡光过程中，该传感器便连续输出标准的脉冲信号。

由电路图可知，将传感器的输出信号经过光耦接到单片机的 T1 口，再设定 T1 为计数器工作方式，这样就可以对所接收到的脉冲进行计数，进而计算出速度里程等。注意一定要将光电传感器的输出信号调好，使其成为标准的脉冲信号，这样 T1 计数器才能正常的工作。再调试的时候可先不接入 EE-SX671，可先用信号发生器产生所需要的脉冲信号来进行模拟，待全部调试好后再接入 EE-SX671。在使用

EE-SX671 之前最好用示波器看一下输出波形是否规整达到要求。一般情况下，输出的就是标准的脉冲信号，如果发现有点偏差可通过外连调理电路将其整理一番。而且最好用电压表的交流档打一下，看一下峰值，是否达到 T1 口的电平识别范围。

EE-SX671 最后要安装到车上，所以事先最好准备一个易弯曲的铁片用螺母将其与 EE-SX671 连接然后在固定于车上，这样可以保证在车子行驶过程中不发生晃动。

4、 液晶显示

由于现在的液晶模块应用已经十分广泛，其使用方法大同小异，所以笔者不准备祥加介绍。大家可根据自己的熟悉程度自行选择显示器件。如果对于液晶不熟悉，完全可以使用数码管作为显示媒介。这里我把自己的液晶显示界面给大家看一下

开机画面

5、 电源

这里我们采用 9V 的电池供电，用 LM7805 进行稳压处理，将 9V 稳降到 5V 以供单片机以及各芯片使用。有条件的可以购买电源模块为系统供电。经测试 9V 电池完全可以满足实际需要。

6、与微机通信

笔者在设计该仪器时，想将所测得的数据传送到电脑中，再用高级语言对数据进行处理，比如速度，在绘制出速度曲线，这样对于教练或是广大自行车爱好者来说，就可以很清晰地看到自己在训练过程中的情况。

这里笔者采用大家熟知的串口通讯方式，采用 MAX232 作为电平转换芯片。相信大家已经很熟悉了，笔者就不赘述了。

四、MCU 软件设计

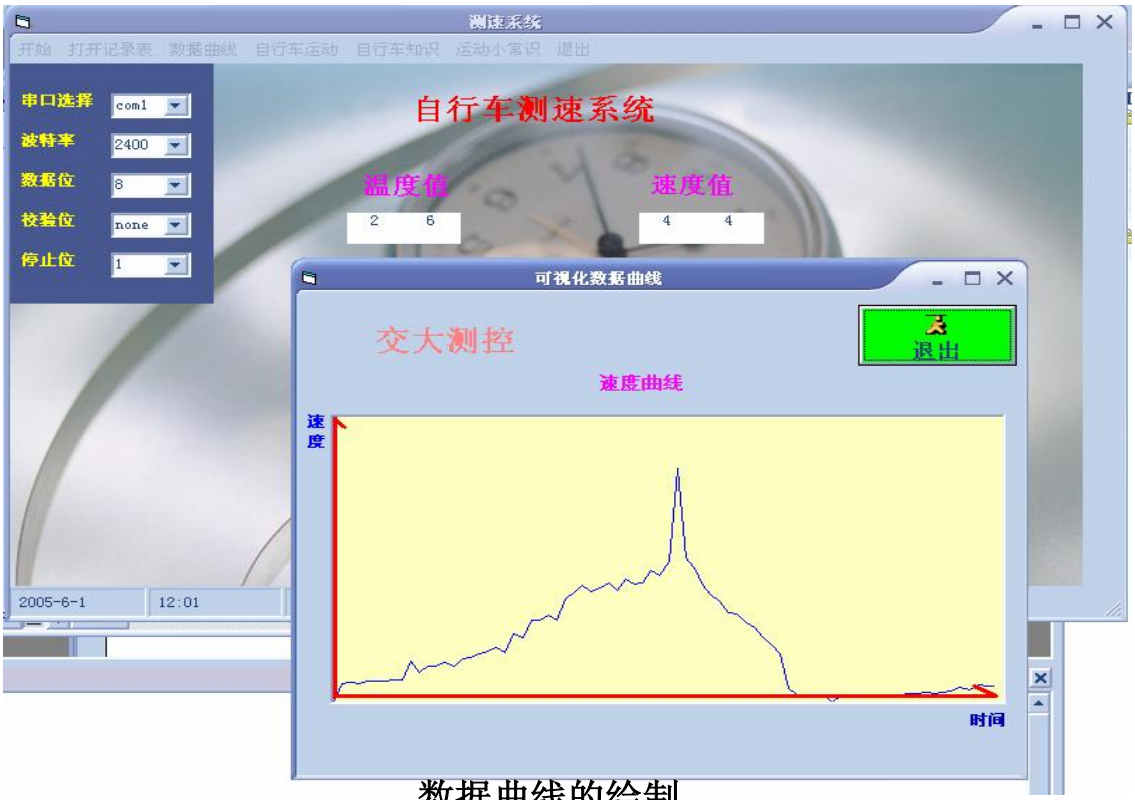
这里采用汇编语言进行编程，软件的任务量比较大，涉及到计速算法，具体会在附录的源程序中详细说明，这里不多述了。

五、上位机软件 bike V1.0 设计

在用高级语言处理上，笔者采用 VB 捆绑数据库将采集得数据转存入数据库中，这样就可以保存每一次的训练数据，以备日后查用。后来索性就编了一个小软件，除了上述功能外，还增加了一些对于自行车运动的常识和知识。

用串口线将测速仪与电脑连接后，即可进行数据的传输。在 PC 机上即可进行数据的接收处理，并可以转存入数据库。

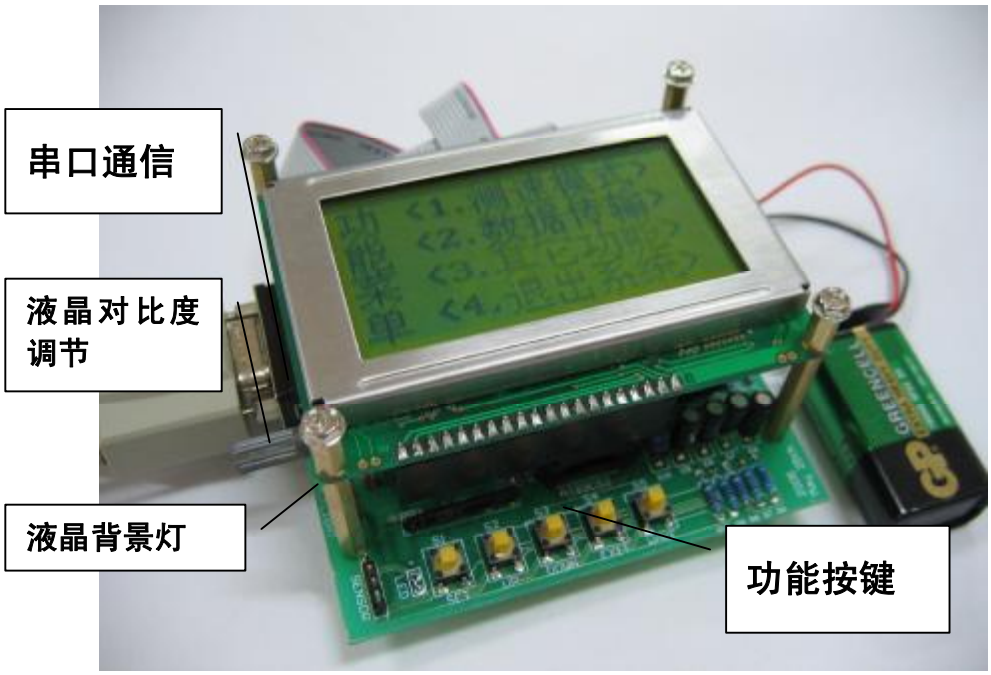
软件主界面



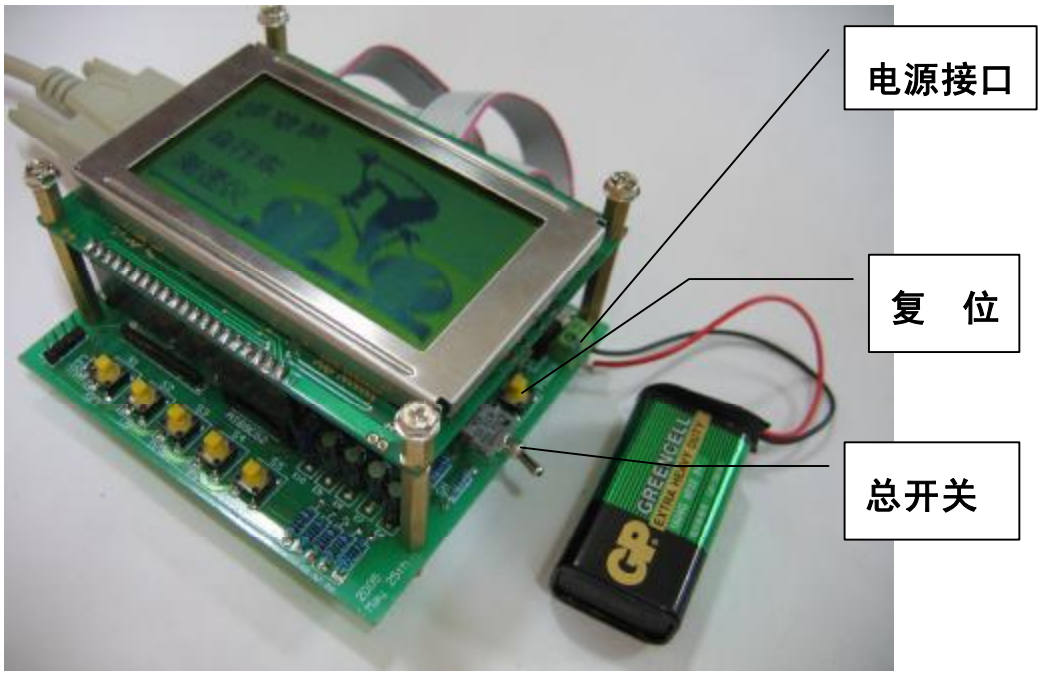
数据曲线的绘制

六、作品实物及测试

产品实物图



总体效果图【1】



总体效果图【2】

本作品采用 9V 电池供电，使用时务必保证电池电量充足，将电池按要求接到系统电源接口处。

拨动开关，使系统工作，将出现如下开机画面：

开机画面

待系统自检 4 秒后便进入系统主菜单，如前图所示。共有四大功能模块：

1. 测速模式：系统的主工作界面，进行速度及相关量的检测
2. 数据传输：将所记录的数据传入 PC 机，再由配套软件 bike v1.0 进行处理，包括绘制曲线及转存入数据库
3. 其它功能：可以进行时间设定及查看版本信息
4. 退出系统：使系统进入节能休眠模式

各功能模块详细测试说明

【注】为方便叙述，将产品的功能按键从左自右依次定义为①号键、②号键、③号键、④号键。

（一）在主菜单下按①号键（SET）即进入<测速模式>，出现如下子菜单：

车型选择

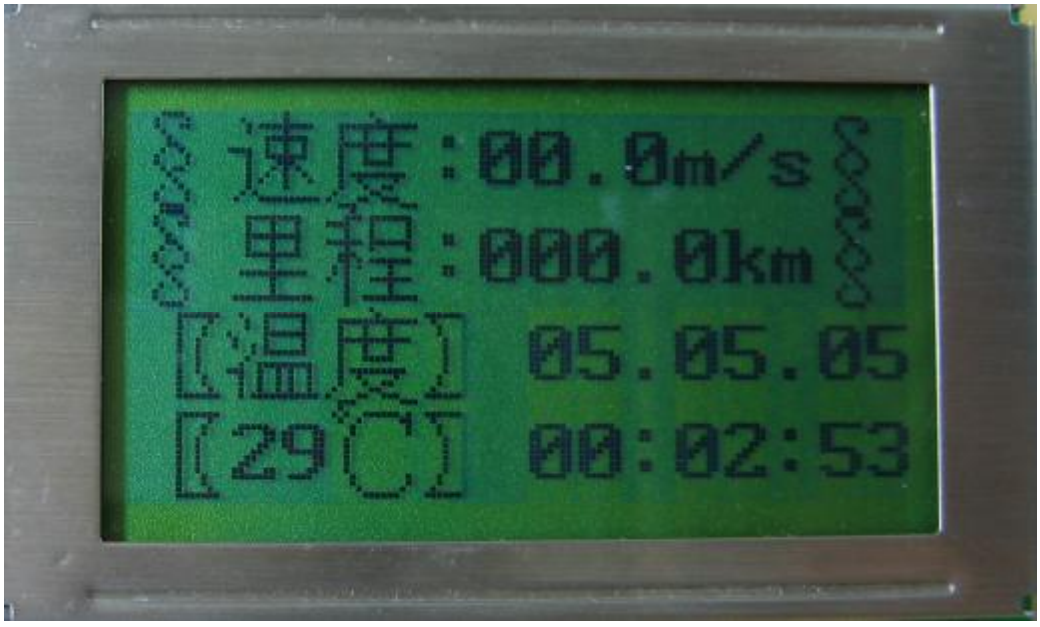
在此子菜单下先进行车型的选择，您可以选择 26 型和 275 型两钟目前最常见的车型进行测量

按③号键可以返回主菜单

如按②号键即选择 275 型，系统会根据您的不同选择为您准确选定参数进行测量。紧接着会出现如下菜单：

采样频率设定

在本菜单中，询问您希望设定的采样频率，这样系统便会按照您的设定时间，每隔固定的时间便向系统内部的 RAM 中自动写入当前的速度值。在选定采样时间后系统便会进入主测量界面：



主测量界面

【注】此时确保您正确接入传感器，否则速度，里程便显示零

在此界面处，随时按下④号键（EXIT），系统便会返回主界面。在采样时间设定处，当设定好时间后，系统会自动将内部存储单元清零。

（二）在主菜单下按②号键即进入<数据传输>，出现如下子菜单：

数据传输菜单

按①号键即进入数据传输模式，在您将测速仪与电脑连接后系统会自动将您刚才运动过程中存储在系统中的速度值传到电脑中以供处理。

按②号键，返回主菜单。

当画面显示【数据传输完毕!】时，表示数据已传输完毕，系统会自动返回主菜单。

【注】在进行数据传输时，为保证正确传输，请务必用串口线将测速仪与电脑连接牢靠。

（三）在主菜单下按③号键即进入<其它功能>，出现如下子菜单：

其它功能菜单

按①号键即进入<时间设定>,出现如下设定画面：

时间设定画面

此时按①号键可以选择修改单元，按②号键（UP），加值；按③号键（DOWN）,减值。修改完毕后，按④号键确定，OK 处光标闪烁反显，后返回上一级菜单。

（四）在主菜单下按④号键即进入<退出系统>，系统即进入节能休眠模式，系统出现如下画面

结束画面

在休眠模式下，按⑤号键（WAKE）即可激活唤醒系统。

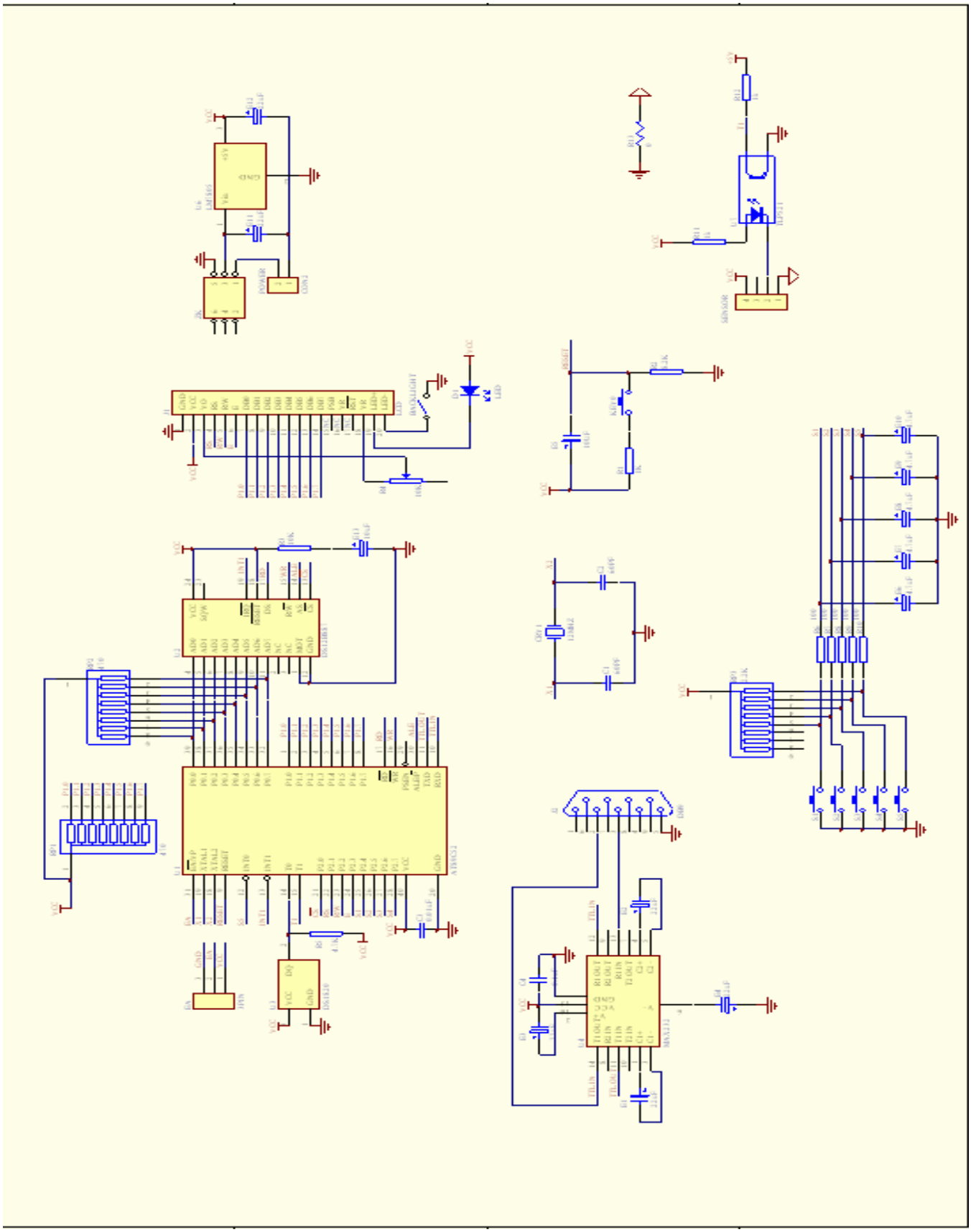
六、结语

以上是对本产品的功能及使用说明。诚然，本次开发的这款产品还有许多亟待改进的地方。包括功能上及软件的设计上。

这是本人大三的时候开发的一个基于 51 单片机的作品，现在看来可以改进的地方很多，比如在体积上，可一选用更小封装的单片机，而且对于手持设备应该低功耗，比如可采用 TI 的 430 系列的单片机，这样就可以用钮扣电池供电，使用时间可以更长。

最后非常感谢本网站能举办这样的活动，这对于普及单片机技术以及提高广大爱好者的动手能力都非常有益！

附录 1 电路



附录 2 源程序

```
;*******总程序*******************
    ;Controller:ST7920
    ;MCU:AT89C52 ,晶体频率：11.0592MHz
    ;LCM:128*64
    ;LCM 型号:带中文字库的 128X64-5ZK
    ;LCM 接口:1:GND 2:VCC 3:V0 4.RS 5:RW 6:E 7--14:DB0-DB7
15:PSB 16:NC 17:RST 18:Vout
;*****************************************************
;***********功能引脚命名*********************************
D_PORT  EQU P1          ;数据口
RS EQU  P2.1        ;液晶指令与数据寄存器设置位
RW  EQU   P2.2     ;液晶读写控制位
E   EQU   P2.3       ;液晶使能端
COM EQU   20H        ;指令寄存器
DAT EQU  21H         ;数据寄存器
TD  EQU  P3.4        ;DS1820 数据引脚
WAKE  EQU  PSW.5      ;液晶复位标志
FX_KEY  EQU PSW.1     ;液晶光标反显开关，FX_KEY=1 一直反显；
FX_KEY=0 反显一次后关闭
```

```
;******************相关使用单元说明*****
    ;25H 光标移动计数器
    ;2EH 用来保护读时钟芯片的 R0
    ;22h 用来存放时钟的 BCD 码
    ;2DH 用来存放修改时间时的汉字查询标志位
    ;2FH 存放温度十位
    ;30H 存放温度各位
    ;85H 用做判断使用内部 RAM 还是时钟 RAM 的标志位，0 为内部，
1 为时钟
    ;37H 用做保存时钟芯片的 RAM 地址
    ;84H 用做保存内部 RAM 的地址
    ;*****************************************************
;****************里程和速度的基本命名******
TAB2  EQU  08H          ;30H 用来查汉字时钟表时使用
LUNZ_Z  EQU  09H        ;31H 中存轮周长整数部分
LUNZ_X  EQU  0AH        ;32H 中存小数部分
SPD_Z  EQU  0BH         ;33H 中存速度的整数部分
SPD_X  EQU  0CH         ;34H 中存小数部分
LC_Z  EQU  0DH          ;35h 中存里程的整数部分
LC_X  EQU  0EH          ;36h 中存小数部分
LC_JW  EQU  0FH         ;37H 中存里程计算时的一个小数向整数部分
进位的标志位
```

```
SPD_SUM EQU 10H        ;38H 中存速度的和（三妙内的）
SPD_PDW EQU 11H        ;39H 中存速度平均值的判断位（三次）
LC_XZB EQU 12H         ;41H 中存里程显示部分的整数的百位
LC_XZS EQU 13H         ;42H 中存里程显示部分的整数的十位
LC_XZG EQU 14H         ;43H 中存里程显示部分的整数的个位
LC_XSF EQU 15H         ;44H 中存里程显示部分的小数的十分位
SPD_XZS EQU 16H        ;45H 中存速度显示部分的整数的十位
SPD_XZG EQ17H          ;46H 中存速度显示部分的整数的个位
SPD_XSF EQU 18H        ;47H 中存速度显示部分的小数的十分位
KM_JW EQU 19H          ;48H 中存里程向千米进位时的标志位
KM EQU 1AH             ;49H 中存里程的公里数
CAI EQU 1BH            ;50H 中是存采样频率的数值
QIAN_Z EQU 1CH         ;51H 中存前一秒时的里程整数。（用于计算
速度）
HOU_Z EQU 1DH          ;52H 中存现在时刻的里程的整数 （用于计
算速度）
QIAN_X EQU 1EH         ;53H 中存前一秒时的里程小数;54H 中存现
在时刻的里程的小数。
HOU_X EQU 1FH          ;54H 中存现在时刻的里程的小数。
QBZW_Z EQU 23H         ;55H 中存里程整数部分前一秒时的进位标志
位
HBZW_Z EQU 24H         ;56H 存整数部分后一秒的进位标志位
```

```
QBZW_X EQU 26H          ;57H 中存里程小数部分前一秒的进位标志位
HBZW_X EQU 27H          ;58H 中存小数部分后一秒的进位标志位
SPD_SUMX EQU 28H        ;59H 中存速度的平均值的小数部分
SDJS EQU 29H            ;61H 中存：轮周长÷每周脉冲数
SD EQU 2AH              ;62H 中存脉冲数和 61h 中相乘后的高八位
TEP EQU 2BH             ;63h 中存的是主程序中判断是否到一分钟的
标志位，用于读温度传感器
CUN EQU 2CH             ;64H 中每到一秒加一，当和 cai 里的数相等
时遍向 ram 中保存一个速度值

;*****************程序地址编排****************

        ORG 0000H
        AJMP    MAIN            ;主程序入口
        ORG 0003H
        LJMP    WAKE_UP         ;INT0 中断入口
        ORG 0013H
        LJMP    SUDU            ;INT1 中断入口
        ORG 001BH
        LJMP    JISUAN          ;T1 中断入口
```

```
;**********************主程序入口地址

                ORG 0040H
MAIN:
                MOV     SP,#60H
                MOV DPTR,#START
                LCALL   PIC_SHOW            ;显示开始画面
                LCALL   TEMPER              ;测试温度
                MOV SPD_Z,#00H
                MOV SPD_X,#00H
                MOV LC_Z,#00H
                MOV LC_X,#00H
                MOV LC_JW,#00H
                MOV SPD_SUM,#00H
                MOV SPD_PDW,#00H
                MOV LC_XZB,#00H
                MOV LC_XZS,#00H
                MOV LC_XZG,#00H
                MOV LC_XSF,#00H
                MOV SPD_XZS,#00H
                MOV SPD_XZG,#00H
                MOV SPD_XSF,#00H
```

```
MOV KM_JW,#00H
MOV KM,#00H
MOV QBZW_Z,#00H
MOV HBZW_Z,#00H
MOV QBZW_X,#00H
MOV HBZW_X,#00H
MOV QIAN_Z,#00H
MOV HOU_Z,#00H
MOV QIAN_X,#00H
MOV HOU_X,#00H
MOV SPD_SUMX,#00H
MOV SD,#00H
MOV TEP,#00H
MOV TAB2,#00H          ;相关单元的初始化
MOV CUN,#00H
 MOV    2FH,#02H
MOV 30H,#09H
 MOV    R1,#85H
MOV @R1,#0
MOV 37H,#0EH
MOV R1,#84H
MOV @R1,#38H
```

```
TIME:
                CLR P2.0            ;时钟芯片的初始化
                MOV R0,#0AH
                MOV A,#70H
                MOVX   @R0,A
                MOV R0,#0BH
                MOV A,#96H
                MOVX   @R0,A
                MOV R0,#0CH
                MOVX   A,@R0        ;基本寄存器的初始化
                MOV R0,#00H
                MOV A,#00H
                MOVX   @R0,A        ;秒单元的初始化
                 MOV    R0,#02H
                MOV A,#00H
                MOVX   @R0,A        ;分单元的初始化
                 MOV    R0,#04H
                MOV A,#00H
                MOVX   @R0,A        ;时单元的初始化
                 MOV    R0,#06H
                MOV A,#05H
                MOVX   @R0,A        ;星期单元的初始化
```

```
                MOV R0,#07H
                MOV A,#05h
                MOVX   @R0,A          ;日单元的初始化
                MOV R0,#08H
                MOV A,#05H
                MOVX   @R0,A          ;月单元的初始化
                MOV R0,#09H
                MOV A,#05H
                MOVX   @R0,A          ;年单元的初始化
                MOV R0,#0AH
                MOV A,#20H
                MOVX   @R0,A
                MOV R0,#0BH
                MOV A,#16H
                MOVX   @R0,A              ;启动时钟芯片

;*****************以上是时钟芯片的初始化
;***********主菜单显示模块**************ZHU_MENU:
                LCALL  INIT_LCM
                MOV COM,#80H
                 MOV    DPTR,#STRING_ZHU
                 LCALL PUT_STRING
```

```
                CLR FX_KEY          ;FX_KEY=0 反显一次后关闭
                clr    EA           ;关中断
                SETB     P2.4
                SETB   P2.5
                 SETB   P2.6
                SETB   P2.7
CHOICE_ZHU:
                JNB      P2.4,CESU_JUMP
                JNB P2.5,CHUANSHU_JUMP
                JNB P2.6,FUNCTION_JUMP
                JNB P2.7,EXIT_JUMP
                AJMP   CHOICE_ZHU
CESU_JUMP:
            LJMP CESU_MENU              ;进入【测速功能】模块
CHUANSHU_JUMP:
            LJMP CHUANSHU_MENU          ;进入【数据传输】模块
FUNCTION_JUMP:
        LJMP     FUNCTION_MENU          ;进入【其它功能】模块
EXIT_JUMP:
            LJMP  EXIT_MENU             ;     【退出系统】
;*****************************************************
```

```
;********************测速菜单
CESU_MENU:
        MOV     R4,#30              ;反显第一行
LCALL FANXIAN               ;反显示主菜单中的【测速模式】项
        LJMP    CESU
;*******************************************************
CESU:
                LCALL  INIT_LCM
                MOV COM,#80H
                MOV DPTR,#STRING_CESU
                LCALL  PUT_STRING       ;显示测速界面
                MOV TMOD,#01100000B         ;设定为方式 2
                MOV TH1,#224
                MOV TL1,#224      ;装初值
                CLR EA
;**********************以上是历程和速度的初始化
                SETB   P2.4
                SETB   P2.5
                SETB   P2.6           ;准备读管脚

JUDGE:
                JNB P2.4,LZC26
```

```
                JNB P2.5,LZC275
                JNB P2.6,RETURN1
                AJMP    JUDGE
;**********************以上是判断按键
LZC26:
                MOV R4,#13              ;反显第二行
  LCALL  FANXIAN              ;反显示测速菜单中的【26 型】项
                MOV LUNZ_Z,#2
                MOV LUNZ_X,#73
                MOV SDJS,#65
                 LJMP   SAMPLE_MENU
LZC275:
                 MOV    R4,#21              ;反显第三行
LCALL  FANXIAN              ;反显示测速菜单中的【275 型】项
                MOV LUNZ_Z,#02H
                MOV LUNZ_X,#193
                MOV SDJS,#68
          LJMP      SAMPLE_MENU
RETURN1:
                MOV R4,#05              ;反显第四行
                LCALL  FANXIAN          ;反显示测速菜单中
```

的【退出】项

```
            LJMP    ZHU_MENU            ;退出本级菜单，返回主
菜单

    ;************************测速菜单结束
    ;**********************数据传输菜单
    CHUANSHU_MENU:
            MOV     R4,#14              ;反显第二行
            LCALL   FANXIAN             ;反显示主菜单中的【数
据传输】项
            LCALL   INIT_LCM            ;重新初始化,显示【数
据传输】菜单项
            MOV     COM,#80H
            MOV     DPTR,#STRING_CHUANSHU
            LCALL   PUT_STRING
                    SETB    P2.4
                    SETB  P2.5
      CHOICE_CHUANSHU:
            JNB     P2.4,BEGIN_CHUAN
            JNB     P2.5,RETURN2
            AJMP    CHOICE_CHUANSHU
    RETURN2: MOV    R4,#14              ;反显第三行
            LCALL   FANXIAN             ;反显示数据传输菜单
```

中的【退出】项

```
        AJMP    ZHU_MENU
BEGIN_CHUAN:
        MOV     R4,#06H             ;反显第二行
        LCALL   FANXIAN             ;反显示数据传输菜单
```

中的【开始传输】项

```
        LCALL   INIT_LCM            ;重新初始化,显示【正
```

在传输....】

```
        MOV     COM,#91H
        MOV     DPTR,#START_CHUAN
        LCALL   PUT_STRING
        LCALL   delay1
        LCALL   delay1
;*******************通信传送模块
        MOV     SCON,#40H                   ;方式 1 传输
        MOV     TMOD,#20H
        MOV     TL1,#0F4H                   ;设置 2.4k 波特
```

率

```
        MOV     TH1,#0F4H
        MOV     PCON,#00H           ;SMOD=0,波特率不加倍
        CLR     TI
        CLR     P2.0
```

```
SETB    TR1                ;T1 开始计时
 MOV R2,#57                ;传送 57 次
 MOV R0,#0EH               ;时钟芯片的内部 RAM 起始
```

单元

```
DO1:
        MOV   SBUF,2FH          ;传送温度的高位
        acall  delay10
         CLR   TI
         MOV   SBUF,30H         ;传送温度的低位
        acall  delay10
         CLR   TI
        MOVX   A,@R0            ;传送速度的整数部分
               MOV  SBUF,A
               acall  delay10
                CLR TI
                INC R0
                MOVX   A,@R0                    ;传送速
```

度的小数部分

```
                 MOV   SBUF,A
                 acall delay10
                 CLR   TI
                 INC   R0
```

```
                DJNZ  R2,DO1

                MOV   R0,#38H          ;单片机的内部 RAM
起始单元
                MOV   R2,#20           ;传送 20 次
    DO:
                MOV   SBUF,2FH         ;传送温度的高位
                acall delay10
                CLR   TI
                MOV   SBUF,30H         ;传送温度的低位
                acall delay10
                CLR   TI
                MOV   SBUF,@R0         ;传送速度的整数部分
                acall delay10
                CLR   TI
                INC   R0
                MOV   SBUF,@R0         ;传送速度的小数部分
                acall delay10
                CLR   TI
                INC   R0
                DJNZ  R2,DO
```

```
                    LCALL INIT_LCM              ;重新初始化,显
```

示【数据传输完毕!】

```
                    MOV   COM,#91H
              MOV     DPTR,#FINISH_CHUAN
              LCALL   PUT_STRING
              LCALL   delay1
              LCALL   delay1
                    LJMP  ZHU_MENU

DELAY10:
                    MOV   R4,#20H
D22:
                    MOV   R5,#248
                    DJNZ  R5,$
                    DJNZ  R4,D22
                    RET

;************************数据传输菜单结束***********


;****************************退出菜单
EXIT_MENU:
```

```
        MOV     R4,#06              ;反显第四行
        LCALL   FANXIAN             ;反显示主菜单中的“退出系统”项
        LCALL   INIT_LCM            ;重新初始化，显示关机画面
        MOV     DPTR,#FINISH
        LCALL   PIC_SHOW            ;显示关机画面
        LCALL   DELAY1

        CLR     WAKE
        MOV     COM,#34H            ;功能设置--8BIT 控制界面，扩充指令集
        LCALL   WRITE_COMMAND
        LCALL   DELAY
                LCALL  DELAY
                MOV COM,#08H            ;进入液晶休眠模式
        LCALL   WRITE_COMMAND
        LCALL   DELAY
                LCALL  DELAY

        MOV     IE,#81H
                SETB   IT0
```

```
        JNB     WAKE,$              ;等待 INT0 中断，判断液晶复位标志 wAkE，wAkE=1,继续执行，wAkE=0,原地执行
        AJMP    ZHU_MENU            ;返回主菜单显示
;*************************退出菜单结束
;*************************采样菜单

SAMPLE_MENU:
        LCALL   INIT_LCM            ;重新初始化,显示【数据传输】菜单项
                MOV COM,#80H
        MOV     DPTR,#STRING_SAMPLE
        LCALL   PUT_STRING

                SETB    P2.4
                SETB  P2.5
        SETB    P2.6
CHAXUN:
                JNB P2.4,LIANG_MIAO
                JNB P2.5,SANSHI_MIAO
                JNB P2.6,LIUSHI_MIAO
                AJMP  CHAXUN
```

```
LIANG_MIAO:
                    MOV R4,#11
                    LCALL  FANXIAN
                    MOV CAI,#2
                    LJMP   CLEAR1

SANSHI_MIAO:
                    MOV R4,#19
                    LCALL  FANXIAN
                    MOV CAI,#30
                    LJMP   CLEAR1

LIUSHI_MIAO:
                    MOV R4,#3
                    LCALL  FANXIAN
                    MOV CAI,#60
                    LJMP   CLEAR1

;************************给所有单元清零

CLEAR1:
```

```
                MOV SPD_Z,#0
                MOV SPD_X,#0
                CLR P2.0

                MOV R1,#85H
                MOV @R1,#0

                MOV 37H,#0EH

                MOV R1,#84H
                MOV @R1,#38H      ;清零的时候重新赋初值

CLEAR:          MOV R1,#85H
                MOV A,@R1         ;85H 用做判断使用内部
RAM 还是时钟 RAM 的标志位，0 为内部，1 为时钟
                CJNE  A,#1,NEIBURAM2

                MOV R0,37H        ;37H用做保存时钟芯片
的 RAM 地址
                CJNE  R0,#128,TIMERAM2
```

```
        MOV R1,#85H
        MOV @R1,#0

        MOV 37H,#0EH

        MOV R1,#84H
        MOV @R1,#38H      ;清零完成后重新赋初值

        LJMP   GZXIANS

TIMERAM2:
        MOV A,SPD_Z
        MOVX   @R0,A
        INC R0
        MOV A,SPD_X
        MOVX   @R0,A
        INC R0
        MOV 37H,R0
        LJMP   BUCUN1
```

```
NEIBURAM2:

                MOV R1,#84H             ;84H 用做保存内部 RAM
的地址
                MOV A,@R1
                MOV R0,A
                CJNE    R0,#60H,NEIBURAM11
                MOV R1,#85H
                MOV @R1,#1
                LJMP    BUCUN1

NEIBURAM11:
                MOV @R0,SPD_Z
                INC R0
                MOV @R0,SPD_X
                INC R0
                MOV R1,#84H
                MOV A,R0
                MOV @R1,A
                LJMP    BUCUN1
BUCUN1:
                 LJMP   CLEAR
```

```
;*************************清零完毕

;*************************采样菜单结束


;*******************************其它功能菜单

FUNCTION_MENU:
                MOV R4,#22              ;反显第三行
                LCALL  FANXIAN          ;反显示主菜单
中的【其它功能】项
FUNC_DOT:
                LCALL  INIT_LCM         ;重新初始化,显
示【其它功能】中的内容
                MOV COM,#80H
          MOV     DPTR,#STRING_FUNCTION
          LCALL   PUT_STRING
          SETB    P2.4
                  SETB  P2.5
          SETB    P2.6
CHOICE_FUNCTION:
           JNB    P2.4,TIME_SET        ;进入【时间设定】
```

```
功能模块
            JNB     P2.5,VERSION_MENU       ;进入【产品信
息】模块
            JNB     P2.6,RETURN3            ;退出本级菜单
            AJMP    CHOICE_FUNCTION
    RETURN3:
            MOV     R4,#03                  ;反显第四行
             LCALL FANXIAN          ;反显示其它功能菜单
中的【退出】项
            AJMP    ZHU_MENU        ;退出本级菜单，返回主菜单
    VERSION_MENU:

            MOV     R4,#19                  ;反显第三行
            LCALL   FANXIAN                 ;反显示其它功能菜单
中的【产品信息】项
            MOV     COM,#80H
            MOV     DPTR,#STRING_VERSION
            LCALL   PUT_STRING
              LCALL     DELAY1
            LCALL   DELAY1
            LCALL   DELAY1
            LCALL   DELAY1
```

```
            AJMP    FUNC_DOT            ;4秒后自动其它功能菜
单

;*******************************其它功能菜单结束
;*******************************时间设定界面

TIME_SET:
            MOV     R4,#11              ;反显第二行
            LCALL   FANXIAN             ;反显[时间设定]项

            SETB    FX_KEY
            CLR     P2.0
                    MOV R0,#0AH             ;向 A 寄存器送命令
                    MOV A,#70H
                    MOVX    @R0,A
                    MOV R0,#0BH             ;向 B 寄存器送命令
                    MOV A,#86H              ;禁止芯片工作，设
定为二进制码格式，24 小时模式
                    MOVX    @R0,A
                    MOV R0,#0CH             ;指向 C 寄存器
                    MOVX    A,@R0               ;读 C 寄存器
```

```
            LCALL   INIT_LCM            ;重新初始化,显示【时
间设定】
            MOV     COM,#80H
            MOV     DPTR,#STRING_TIME
            LCALL   PUT_STRING

    TIME_DISP:
                    MOV TAB2,#00H
            MOV     COM,#90H
                    LCALL  WRITE_COMMAND

                    MOV R0,#09H
            LCALL   BIN_BCD
                    LCALL  TDISP
                    LCALL  HANZI

                    MOV R0,#08H
                    LCALL   BIN_BCD
            LCALL   TDISP
            LCALL   HANZI

                    MOV R0,#07H
```

```
LCALL   BIN_BCD
LCALL   TDISP
LCALL   HANZI

        MOV R0,#06H
        LCALL   BIN_BCD
        LCALL  TDISP
        LCALL  HANZI

        MOV R0,#04H
        LCALL   BIN_BCD
        LCALL  TDISP
        LCALL  HANZI

        MOV R0,#02H
        LCALL   BIN_BCD
        LCALL  TDISP
        LCALL  HANZI

        MOV R0,#00H
        LCALL   BIN_BCD
        LCALL  TDISP
```

```
                LCALL  HANZI
                MOV COM,#9FH
          MOV       DPTR,#STRING_OK
          LCALL     PUT_STRING

;******************以上界面显示**********************


          MOV     R4, #04
          MOV     25H,#07       ;光标移动计数器，当为 0 时，
```

要重新复位

```
          SETB    P2.4
          SETB    P2.5
          SETB    P2.6
           SETB    P2.7

           LCALL  FANXIAN
FAN:
           JNB     P2.4,xiaodou1
          ajmp    j2
xiaodou1:mov     r0,#12
```

```
xiaodou11:
          lcall  delay
         djnz    r0,xiaodou11
          JNB    P2.4,gb_shift

 j2:      JNB    P2.5,xiaodou2
         ajmp    j3
xiaodou2:mov     r0,#10
xiaodou22:
         lcall   delay
         djnz    r0,xiaodou22
         JNB    P2.5,time_add

j3:      JNB    P2.6, xiaodou3
         ajmp    j4
xiaodou3:mov     r0,#10
xiaodou33:
         lcall   delay
         djnz    r0,xiaodou33
         JNB    P2.6,time_minus
```

```
j4:     JNB     P2.7,TIME_OVER
        AJMP    FAN

;*****************光标移动
GB_SHIFT:
        DJNZ    25H,LEFT_SHIFT
        MOV     25H,#07
          MOV     R4,#12
GB_FUWEI:

        MOV     COM,#14H        ;设定液晶的光标向右移动
        LCALL   WRITE_COMMAND
        LCALL   DELAY
        LCALL   DELAY
        DJNZ    R4,GB_FUWEI             ;R4为光标移动位数
        AJMP    FAN
LEFT_SHIFT:
        MOV     R4,#02
        LCALL   FANXIAN
          AJMP    FAN
;*****************************************************
```

```
TIME_ADD:
                MOV R2,#01H
                ACALL  UPDATE
TIME_MINUS:
                MOV R2,#00H
                ACALL  UPDATE
TIME_OVER:
                CLR P2.0
                MOV R0,#0AH
                MOV A,#20H
                MOVX   @R0,A
                MOV R0,#0BH
                MOV A,#16H
                MOVX   @R0,A            ;启动时钟芯片

                CLR FX_KEY
                MOV COM,#02H            ;光标重新回到原点
                LCALL  WRITE_COMMAND
                LCALL  DELAY
                LCALL  DELAY
                MOV R4,#32              ;设定光标向又移动 32 次
TIME_OVER1:
```

```
                MOV COM,#14H      ;设定液晶的光标向右移动
        LCALL   WRITE_COMMAND
        LCALL   DELAY
                LCALL  DELAY
        DJNZ    R4,TIME_OVER1       ;R4 为光标移动位数
        mov     r4,#01
                lcall  fanxian
        AJMP    ZHU_MENU

UPDATE:
                MOV R1,25H
                CJNE   R1,#07,MINUTE
        LCALL   BUSY
        MOV     R0,#00H
        MOVX    A,@R0
                mov r6,#9ch
                CJNE   R2,#01H,DEC1
                cjne    a,#59,inc1
                mov a,#00
                ajmp    update1
inc1:           INC A
        ajmp    update1
```

```
DEC1:           cjne   a,#00,d1
                mov    a,#59
                ajmp   update1
d1:             DEC A
        ajmp    update1
      MINUTE:
                CJNE   R1,#06H,HOUR
                LCALL  BUSY
        MOV     R0,#02H
        MOVX    A,@R0
                mov r6,#9ah
                CJNE   R2,#01H,DEC2
                cjne   a,#59,inc2
                mov a,#00
                ajmp   update1
inc2:           INC A
        ajmp    update1
DEC2:           cjne   a,#00,d2
        mov     a,#59
        ajmp    update1
d2:             DEC A
                ajmp   update1
```

```
HOUR:
                CJNE   R1,#05H,WEEK
                LCALL  BUSY
        MOV     R0,#04H
        MOVX    A,@R0
                mov r6,#98h
                CJNE   R2,#01H,DEC3
                cjne    a,#23,inc3
                mov a,#00
                ajmp    update1
inc3:           INC A
        ajmp    update1
DEC3:           cjne    a,#00,d3
        mov     a,#23
        ajmp    update1
d3:             DEC A
        ajmp    update1

WEEK:
                CJNE   R1,#04H,DAY
                LCALL  BUSY
```

```
                MOV   R0,#06H
               MOVX   A,@R0
               mov r6,#96h
               CJNE   R2,#01H,DEC4
               cjne   a,#7,inc4
                mov   a,#01
               ajmp    update1
inc4:          INC A
               ajmp    update1
DEC4:          cjne   a,#01,d4
                mov     a,#07
                ajmp    update1
d4:            DEC A
               ajmp    update1

DAY:
               CJNE   R1,#03H,MONTH
               LCALL  BUSY
                MOV   R0,#07H
                MOVX  A,@R0
               mov r6,#94h
               CJNE   R2,#01H,DEC5
```

```
                cjne   a,#31,inc5
                mov      a,#01
                ajmp     update1
inc5:           INC A
                ajmp     update1

DEC5:           cjne   a,#01,d5
                 mov   a,#31
                ajmp   update1
d5:              DEC   A
                ajmp     update1

MONTH:
                CJNE   R1,#02H,YEAR
                LCALL  BUSY
                 MOV   R0,#08H
                MOVX   A,@R0
                mov r6,#92h
                CJNE   R2,#01H,DEC6
                cjne     a,#12,inc6
                mov a,#01
                ajmp   update1
```

```
inc6:       INC A
            ajmp    update1
DEC6:       cjne   a,#01,d6
             mov   a,#12
            ajmp    update1
d6:         DEC A
            ajmp    update1

YEAR:
            LCALL  BUSY
             MOV   R0,#09H
            MOVX   A,@R0
            mov r6,#90h
            CJNE   R2,#01H,DEC7
            INC A
            ajmp    update1
DEC7:       cjne    a,#00,d7
            mov     a,#00
            ajmp    update1
d7:         DEC A
            ajmp   update1
```

```
update1:

                MOVX   @R0,A
                MOV COM,r6
                LCALL  WRITE_COMMAND
                LCALL    BIN_BCD
                LCALL  TDISP
                MOV R4,#01
                LCALL  FANXIAN
                AJMP   FAN

;*****************************时间设定界面结束
;*****************************工作显示界面
GZXIANS:
                LCALL  INIT_LCM

                SETB   IT1
                SETB   EA
                SETB   ET1
                SETB   TR1    ;开 T1 中断，并开始计数
                SETB   EX1
                SETB   PX1              ;开 INT1 中断
```

```
GZXIANS0:
                MOV A,TEP          ;TEP 中存一个是否到一分
中的标志
                CJNE   A,#30,GZXIANS1
                MOV TEP,#00H       ;到一分钟后就读温度传感
器
                LCALL  TEMPER

GZXIANS1:
;************************第一行
                MOV COM,#80H
                MOV DPTR,#HANG1
                LCALL  PUT_STRING       ;显示“$速度:”

                MOV A,SPD_XZS
                MOV DPTR,#SHU
                MOVC   A,@A+DPTR
                MOV DAT,A
                LCALL  WRITE_DATA       ;显示速度十位

                MOV A,SPD_XZG
```

```
MOV DPTR,#SHU
MOVC    A,@A+DPTR
MOV DAT,A
LCALL   WRITE_DATA          ;显示速度个位

MOV A,#10               ;显示小数点
MOV DPTR,#SHU
MOVC    A,@A+DPTR
MOV DAT,A
LCALL   WRITE_DATA

MOV A,SPD_XSF
MOV DPTR,#SHU
MOVC    A,@A+DPTR
MOV DAT,A
LCALL   WRITE_DATA      ;显示速度的十分位

MOV A,#11
MOV DPTR,#SHU
MOVC    A,@A+DPTR
MOV DAT,A
LCALL   WRITE_DATA          ;显示 m
```

```
                MOV A,#12
                MOV DPTR,#SHU
                MOVC    A,@A+DPTR
                MOV DAT,A
                LCALL   WRITE_DATA      ;显示/

                MOV A,#13
                MOV DPTR,#SHU
                MOVC    A,@A+DPTR
                MOV DAT,A
                LCALL   WRITE_DATA      ;显示 s

;************************第三行
                MOV DPTR,#HANG3
                LCALL   DISP_STR_LOOP       ;显示“(温度):”

                MOV R0,#09H
                LCALL    BIN_BCD
                LCALL   TDISP           ;显示年里的数

                MOV A,#10
                MOV DPTR,#SHU
```

```
                MOVC    A,@A+DPTR
                MOV DAT,A
                LCALL   WRITE_DATA      ;显示.

                MOV R0,#08H
                LCALL    BIN_BCD
                LCALL   TDISP           ;显示月里的数

                MOV A,#10
                MOV DPTR,#SHU
                MOVC    A,@A+DPTR
                MOV DAT,A
                LCALL   WRITE_DATA      ;显示.

                MOV R0,#07H
                LCALL    BIN_BCD
                LCALL   TDISP           ;显示日里的数

;************************第二行

                MOV DPTR,#HANG2
                LCALL   DISP_STR_LOOP       ;显示“$里程:”
```

```
MOV A,LC_XZB
MOV DPTR,#SHU
MOVC    A,@A+DPTR
MOV DAT,A
LCALL   WRITE_DATA          ;显示里程的百位

MOV A,LC_XZS
MOV DPTR,#SHU
MOVC    A,@A+DPTR
MOV DAT,A
LCALL   WRITE_DATA          ;显示里程的十位

MOV A,LC_XZG
MOV DPTR,#SHU
MOVC    A,@A+DPTR
MOV DAT,A
LCALL   WRITE_DATA          ;显示里程的个位

MOV A,#10               ;显示小数点
MOV DPTR,#SHU
MOVC    A,@A+DPTR
MOV DAT,A
```

```
        LCALL  WRITE_DATA

        MOV A,LC_XSF
        MOV DPTR,#SHU
        MOVC   A,@A+DPTR
        MOV DAT,A
        LCALL  WRITE_DATA       ;显示里程的十分位

        MOV A,#14
        MOV DPTR,#SHU
        MOVC   A,@A+DPTR
        MOV DAT,A
        LCALL  WRITE_DATA       ;显示 K

        MOV A,#15
        MOV DPTR,#SHU
        MOVC   A,@A+DPTR
        MOV DAT,A
        LCALL  WRITE_DATA       ;显示 M

;************************第四行
```

```
MOV DPTR,#HANG4
LCALL  DISP_STR_LOOP
 MOV    A,2FH
MOV DPTR,#SHU
MOVC   A,@A+DPTR
MOV DAT,A
LCALL  WRITE_DATA
MOV A,30H
MOV DPTR,#SHU
MOVC   A,@A+DPTR
MOV DAT,A
LCALL  WRITE_DATA       ;显示温度的数值
MOV DPTR,#HANG5
LCALL  DISP_STR_LOOP
 MOV    R0,#04H
LCALL    BIN_BCD
LCALL  TDISP            ;显示时里的数

MOV A,#16
MOV DPTR,#SHU
MOVC   A,@A+DPTR
MOV DAT,A
```

```
                LCALL  WRITE_DATA       ;显示:

                MOV R0,#02H
                LCALL    BIN_BCD
                LCALL  TDISP            ;显示分里的数

                MOV A,#16
                MOV DPTR,#SHU
                MOVC   A,@A+DPTR
                MOV DAT,A
                LCALL  WRITE_DATA       ;显示:

                MOV R0,#00H
                LCALL    BIN_BCD
                LCALL  TDISP            ;显示秒里的数
                SETB   P2.7
;**********;查询按键。若有则返回主菜单

                JNB     P2.7,xiao4
                LJMP   GZXIANS0
xiao4:
                mov     r0,#10
```

```
xiao44:
                lcall  delay
                djnz   r0,xiao44
                 JNB     P2.7,TIAO
;*************;查询按键。若有则返回主菜单
                LJMP   GZXIANSO
TIAO:
                mov     r0,#100
xiao55:
                lcall  delay
                jnz r0,xiao55
                LJMP   ZHU_MENU

;**************************工作显示界面结束
;**************************【里程中断程序】

JISUAN:
                MOV A,LUNZ_Z
YICHU1:         CLR C
                ADD A,LC_Z
                MOV LC_Z,A              ;计算里程的整数部分
                JC  KM_CHULI
```

```
                AJMP    NOJINW1

KM_CHULI:
                INC KM_JW
                INC HBZW_Z              ;增加 HBZW_Z 中的数，
以后计算速度时用

                MOV A,HBZW_Z
                CJNE    A,#255,KM_CHULI1
                MOV A,#255
                CLR C
                SUBB    A,QBZW_Z
                MOV HBZW_Z,A
                MOV QBZW_Z,#0           ;标志位溢出时的处理

KM_CHULI1:
                MOV A,KM_JW
                CJNE    A,#04H,NOJINW1      ;判断是否该向
公里处进 1
                MOV KM_JW,#00H
                INC KM
                MOV A,#24
```

```
                AJMP    YICHU1

NOJINW1:
                MOV A,LUNZ_X
YICHU2:         CLR C
                ADD A,LC_X              ;计算里程的小数部分
                MOV LC_X,A
                JC  LC_CHULI        ;看是否有溢出
                AJMP    NOJINW2

LC_CHULI:
                INC LC_JW
                INC HBZW_X              ;增加 HBZW_X 中的数，
```

计算速度时用

```
                MOV A,HBZW_X
                CJNE    A,#255,LC_CHULI1
                MOV A,#255
                CLR C
                SUBB    A,QBZW_X
                MOV HBZW_X,A
                MOV QBZW_X,#0           ;标志位溢出时的处理
```

```
LC_CHULI1:
                MOV A,LC_JW
                CJNE    A,#04H,HUIJIA       ;判断小数部分是否进四次位，是否应该向整数部分进 1

                MOV LC_JW,#00H
                MOV A,LC_Z
                CLR C
                ADD A,#01H              ;加一以后再判断整数部分是否有溢出
                MOV LC_Z,A
                JNC HUIJIA
PANDUAN:
                INC KM_JW
                INC HBZW_Z              ;增加 HBZW_Z 中的数，以后计算速度时用
                MOV A,KM_JW
                CJNE    A,#04H,HUIJIA       ;判断是否该向公里处进 1
                MOV KM_JW,#00H
                INC KM
                MOV A,#24
```

```
                CLR C
                ADD A,LC_Z
                MOV LC_Z,A
                JC  PANDUAN

    HUIJIA:
                MOV A,#24          ;每四次进位，256×4＝
1024，除去 1000，还有 24
                AJMP   YICHU2              ;返回判断是否小数
部分有溢出
    NOJINW2:
                MOV A,KM
                MOV B,#100
                DIV AB
                MOV LC_XZB,A
                MOV A,B
                MOV B,#10
                DIV AB
                MOV LC_XZS,A
                MOV A,B
                MOV LC_XZG,A       ;将公里单元里的整数分成
三个 BCD 码
```

```
                MOV A,LC_Z
                MOV B,#10
                DIV AB
                MOV LC_XSF,A
                MOV A,KM_JW
                MOV B,#25
                MUL AB
                ADD A,LC_XSF
                MOV B,#10
                DIV AB
                MOV LC_XSF,A

                RETI


;*****************************里程中断程序结束
;*****************************时钟中断程序（计算速度）
SUDU:
                MOV R0,#0CH
                MOVX   A,@R0          ;消去相应的时钟标志
位
                INC TEP               ;增加温度标志位中的数，
```

为读温度传感器做标志

```
;*****************
                INC CUN
                MOV A,CUN
                CJNE    A,CAI,BUCUN
                MOV CUN,#00H
                CLR P2.0

                MOV R1,#85H
                MOV A,@R1           ;85H 用做判断使用内部
RAM 还是时钟 RAM 的标志位，0 为内部，1 为时钟
                CJNE    A,#1,NEIBURAM

                MOV R0,37H          ;37H用做保存时钟芯片
的 RAM 地址
                CJNE    R0,#128,TIMERAM
                LJMP    BUCUN

    TIMERAM:
                MOV A,SPD_Z
                MOVX    @R0,A
```

```
                INC R0
                MOV A,SPD_X
                MOVX   @R0,A
                INC R0
                MOV 37H,R0
                LJMP   BUCUN

NEIBURAM:

                MOV R1,#84H          ;84H 用做保存内部 RAM
```

的地址

```
                MOV A,@R1
                MOV R0,A
                CJNE   R0,#60H,NEIBURAM1
                MOV R1,#85H
                MOV @R1,#1
                LJMP   BUCUN

NEIBURAM1:

                MOV @R0,SPD_Z
                INC R0
                MOV @R0,SPD_X
```

```
                INC R0
                MOV R1,#84H
                MOV A,R0
                MOV @R1,A
                LJMP    BUCUN

;******************

BUCUN:
                MOV A,TL1           ;读出现在时刻的脉冲数，
```

计算出相应的路程使得速度精确

```
                MOV TL1,#224        ;同时重装初值，以免里程
```

多计算

```
                CLR C
                SUBB    A,#224
                MOV B,SDJS
                MUL AB
                MOV SD,B
                CLR C
                ADD A,LC_X              ;将相乘后的低八位加
```

到小数部分

```
                MOV LC_X,A
                JNC NOJINW3          ;判断是否溢出，无就直
接处理相乘后的高八位
                AJMP   YICHU4        ;有溢出就到YICHU4

NOJINW3:                             ;处理相乘后的高八位
                MOV A,SD
                CJNE   A,#00H,YUNSUAN    ;高八位表示相
乘后有多少个256，每四个就应该向整数进一
                AJMP   TIAOCHU

YUNSUAN:
                DEC SD
YICHU4:         INC LC_JW
                INC HBZW_X
                MOV A,LC_JW
                CJNE   A,#04H,NOJINW3
                MOV LC_JW,#00H
                MOV A,LC_Z
                CLR C
                ADD A,#01H
                JNC HUIJIA1
```

```
YICHU3:
        INC KM_JW
        INC HBZW_Z          ;增加 HBZW_Z 中的数，
以后计算速度时用
        MOV A,KM_JW
        CJNE   A,#04H,HUIJIA1     ;判断是否该向
公里处进 1
        MOV KM_JW,#00H
        INC KM
        MOV A,#24
        CLR C
        ADD A,LC_Z
        MOV LC_Z,A
        JC  YICHU3

HUIJIA1:
        MOV A,#24
        CLR C
        ADD A,LC_X
        MOV LC_X,A
        JC  YICHU4
```

```
                AJMP    NOJINW3

;*****************以上是处理没有进入中断计算内的里程
;**************以下是计算速度

TIAOCHU:
;***************以下是计算速度的整数部分

                MOV A,LC_Z          ;把现在时刻里程整数部分给 a
                MOV HOU_Z,LC_Z      ;把现在时刻里程整数部分给 HOU_Z
                CLR C
                SUBB    A,QIAN_Z    ;用现在时刻的里程减去前一秒时刻的里程
                MOV QIAN_Z,HOU_Z        ;把现在时刻的里程个前一秒时刻的里程，准备下次使用
                MOV SPD_Z,A         ;把减的值给数度的整数部分

                MOV A,HBZW_Z
                SUBB    A,QBZW_Z
```

```
                MOV QBZW_Z,HBZW_Z       ;通过标志位判断在
```

这一秒中里程溢出了几次，一次就是 256 米

```
                MOV B,#255
                MUL AB
                ADD A,SPD_Z             ;修正速度整数部分的
```

值

```
                MOV SPD_Z,A

    ;**************以下是计算速度的小数部分
                MOV A,LC_X              ;把现在时刻里程小数
```

部分给 a

```
                MOV HOU_X,LC_X          ;把现在时刻里程小数
```

部分给 HOU_Z

```
                CLR C
                SUBB    A,QIAN_X        ;用现在时刻的里程减
```

去前一秒时刻的里程

```
                MOV QIAN_X,HOU_X        ;把现在时刻的里程
```

个前一秒时刻的里程，准备下次使用

```
                MOV B,#10
                DIV AB
                MOV SPD_X,A
```

```
                MOV A,HBZW_X
                SUBB    A,QBZW_X
                MOV QBZW_X,HBZW_X       ;通过标志位判断在
```

这一秒中里程溢出了几次，一次就是 0.256 米

```
                MOV B,#25
                MUL AB
                ADD A,SPD_X
                MOV B,#100
                DIV AB
                ADD A,SPD_Z
                MOV SPD_Z,A
                MOV A,B
                MOV B,#10
                DIV AB
                MOV SPD_X,A             ;修正速度小数部分的
```

值

```
    ;***************以下是计算三秒内的速度的平均值

                MOV A,SPD_Z
                ADD A,SPD_SUM
                MOV SPD_SUM,A
```

```
                MOV A,SPD_X
                ADD A,SPD_SUMX
                MOV SPD_SUMX,A

                INC SPD_PDW          ;将速度求和，并判断是否到了三次
                MOV A,SPD_PDW
                CJNE   A,#03H,FANHUI      ;没到三秒就返回
                MOV SPD_PDW,#00H          ;到三次后，将标志位重置位 0
                MOV A,SPD_SUM
                MOV B,#03H
                DIV AB               ;求出平均值
                MOV SPD_SUM,A        ;先将整数部分暂存在 SPD_SUM 中
                MOV A,B              ;将余数部分的值给 a
                MOV B,#10
                MUL AB
                ADD A,SPD_SUMX       ;将余数乘以 10 和原来的小数部分相加，为产生小数作准备
```

```
                MOV SPD_SUMX,A


                MOV A,SPD_SUMX
                MOV B,#3            ;将小数部分除以 3
                DIV AB
                MOV B,#10
                DIV AB
                ADD A,SPD_SUM       ;十位有 1 就应该进
                MOV SPD_SUM,A
                MOV A,B
                MOV SPD_XSF,A       ;将平均值的小数部分
```

给显示单元

```
                MOV A,SPD_SUM       ;将速度的整数部分重
```

新给 a

```
                MOV B,#0AH
                DIV AB
                MOV SPD_XZS,A
                MOV A,B
                MOV SPD_XZG,A       ;将速度的整数部分分
```

位 BCD 码，并存入相应的显示单元

```
                MOV SPD_SUM,#00H
                MOV SPD_SUMX,#00H
FANHUI:
                RETI

;************************时钟中断程序结束
;########################以下是一些相关的子程序#
;****************************温度子程序
TEMPER:
                mov 2Fh,#00h     ;保存转换结果高位清零
                mov 30h,#00h     ;保存转换结果低位清零
                mov a,#00h

                setb TD
                clr ea
gongzuo:
                Lcall reset
                mov a,#0cch
                lcall write
                mov a,#44h
                lcall write
                mov 31h,#78
```

```
                lcall delay5    ;等待 98ms 温度转换结束

                Lcall reset     ;重新复位
                mov a,#0cch
                lcall write
                mov a,#0beh     ;送读暂存器命令
                lcall write
                lcall read
                mov 30h,a
                lcall read
                mov 2Fh,a
                mov a,#0fh
                anl a,2Fh
                swap a
                mov b,a
                mov a,#0f0h
                anl a,30h
                swap a
                orl a,b         ;把得到的结果处理后保存
到 acc
                rlc a           ;下面是对温度的正负值进行
处理
```

```
                jc fushu
                rrc a
bcdchange:
                mov b,#100
                div ab
                mov 32h,a
                mov a,b
                mov b,#10
                div ab
                mov 2Fh,a
                mov a,b
                mov 30h,a
                ret

fushu:                  ;温度为负时，把补码转换回来
                setb c
                rrc a
                cpl a
                add a,#01h          ;取反加一
                Ljmp bcdchange

reset:                  ;复位子程序
```

```
                clr TD
                mov 35h,#240
                djnz 35h,$
                setb TD          ;释放总线
                mov 35h,#40
                djnz 35h,$       ;等待 80us
                orl C,TD
                jC  reset     ;判断是否有从设备，没有则再
次复位
    wait:
                mov 35h,#120     ;等待 240us 从设备释放总
线
                djnz 35h,$
                orl C,TD
                jnc wait
                setb  ea
                ret

    write:
                mov r4,#8        ;写入 ds1820 的 bit 数,一
```

```
个字节 8 个 bit
    write1:     setb TD
                mov 35h,#6
                rrc a          ;把一个字节 data(A)分成 8 个
bit 环移给 C
                clr TD         ;开始写入 ds1820 总线要
处于复位(低)状态
    write2:     djnz 35h,$     ;ds1820 总线复位保持
12us
                mov TD,C       ;写入一个 bit
                mov 35h,#26
    write3:
                djnz 35h,$     ;等待 52us
                djnz r4,write1     ;写入下一个 bit
                setb TD        ;重新释放 ds1820 总线
                ret

    read:
                mov r3,#8      ;连续读 8 个 bit
    re1:        clr TD         ;读前总线保持为低
                mov 35h,#4
                setb TD        ;开始读，总线释放
```

```
re2:        djnz 35h,re2        ;持续 8us
            mov c,TD      ;从 ds1820 总线读得一个 bit
            rrc A         ;把读得的位值环移给 A
            mov 35h,#30
re3:        djnz 35h,re3        ;持续 60us
            djnz r3,re1     ;读下一个 bit
            setb TD       ;重新释放 ds1820 总线

            SETB   EA     ;返回时重新打开所以的中断
            ret


delay5:
loop02:     mov 33h,#60
loop01:     mov 34h,#40
            djnz 34h,$
            djnz 33h,loop01
            djnz 31h,loop02
            ret

;*****************************温度子程序结束
```

```
;************************图形显示子程序
PIC_SHOW:
                    LCALL INIT_LCM
                    MOV     COM,#32H            ;功能设置---8BIT 控制界面，绘图显示 ON,扩充指令集
                LCALL   WRITE_COMMAND       ;调用写指令子程序
                LCALL   DELAY               ;延迟 39uS
                MOV     COM,#32H            ;功能设置---8BIT 控制界面，绘图显示 ON
                LCALL   WRITE_COMMAND       ;调用写指令子程序
                LCALL   DELAY               ;延迟 39uS
                MOV     COM,#36H            ;功能设置---8BIT 控制界面，扩充指令集
                LCALL   WRITE_COMMAND       ;调用写指令子程序
                LCALL   DELAY               ;延迟 39uS

DISPLAY6:

              MOV     R2,#32                ;32 行,(双屏结构中上半屏)
```

```
        MOV     R3,#80H             ;Y 地址寄存器

    DISP6:
        MOV     COM,R3              ;设置绘图区的 Y 地址
坐标
        INC     R3                  ;Y 地址加 1
        LCALL   WRITE_COMMAND
        MOV     COM,#80H            ;设置绘图区的 X 地址
坐标
        LCALL   WRITE_COMMAND
        MOV     R1,#16              ;16*8 列

    DISP7:
        CLR  A
        MOVC    A,@A+DPTR
        MOV     DAT,A
        LCALL   WRITE_DATA
        INC     DPTR
        DJNZ    R1,DISP7
        DJNZ    R2,DISP6            ;写满全屏的 16*8 字节
X64
        MOV     R2,#32              ;32 行,(双屏结构的下
```

```
半屏)
        MOV     R3,#80H             ;Y 地址寄存器

    DISP8:
        MOV     COM,R3              ;设置绘图区的 Y 地址
坐标
        INC     R3                  ;Y 地址加 1
        LCALL   WRITE_COMMAND
        MOV     COM,#88H            ;设置绘图区的 X 地址
坐标
        LCALL   WRITE_COMMAND
        MOV     R1,#16              ;16*8 列

    DISP9:
        CLR  A
        MOVC    A,@A+DPTR
        MOV     DAT,A
        LCALL   WRITE_DATA
        INC     DPTR
        DJNZ    R1,DISP9
        DJNZ    R2,DISP8            ;写满全屏的 16*8 字节
X64
```

```
        LCALL   DELAY1                  ;1S 延时子程序
                LCALL   DELAY1
                LCALL   DELAY1
                LCALL   DELAY1
                ;MOV   R4,#10
                RET


;*****************图形显示子程序结束
;************************反显菜单子程序
;******入口参数为 R4，存放光标移动位数***************
FANXIAN:
GBXW:           MOV COM,#10H            ;设定液晶的光标向
左移动
        LCALL   WRITE_COMMAND
        LCALL   DELAY
        LCALL   DELAY
        DJNZ    R4,GBXW             ;R4 为光标移动位数

        MOV     COM,#0DH            ;将光标所在位置反显
        LCALL   WRITE_COMMAND
        LCALL   DELAY
                LCALL  DELAY
```

```
                JB      FX_KEY,FANXIAN_OVER

        LCALL   DELAY1              ;反显 1 秒


        MOV     COM,#0CH            ;光标处恢复正常显示
        LCALL   WRITE_COMMAND
        LCALL   DELAY
                LCALL  DELAY
FANXIAN_OVER:
                RET
;***********************反显菜单子程序结束
;*********INT0 中断,等待外部触发信号,激活液晶显示*******
WAKE_UP:
        SETB    WAKE                ; 使 液 晶 复 位 标 志
wAkE=1
        MOV     COM,#0CH            ;进入液晶工作模式
        LCALL   WRITE_COMMAND
        LCALL   DELAY
                LCALL  DELAY
        RETI
```

```
;**************延时 39us 子程序
DELAY:   MOV      R7,#0FFH

DELAY4:  MOV      R6,#00FH

LOOP2:   DJNZ     R6,LOOP2

         DJNZ     R7,DELAY4

         RET
;**************延时 1s 子程序
DELAY1:  MOV      R4,#08H

DELAY11:MOV       R7,#0FFH

DELAY41:MOV   R6,#0FFH

         DJNZ R6,$

         DJNZ     R7,DELAY41

          DJNZ     R4,DELAY11
```

```
        RET

XX:     MOV     R5,#0FFH

XX1:    MOV   R6,#0FFH

XX2:    DJNZ R6,XX2

        DJNZ    R5,XX1

        RET
```

;*****************************以上是延时子程序

;******** 液 晶 初 始 化 模 块 子 程 序

```
INIT_LCM:                           ;液晶初始化程序
          MOV     COM,#30H          ;功能设置---8BIT控制界面，基本指令集
          ACALL   WRITE_COMMAND     ;调用写指令子程序
          LCALL   DELAY             ;延迟 39uS
          MOV     COM,#0CH          ;显示打开，光标关，反白显示关
```

```
                ACALL   WRITE_COMMAND       ;调用写指令子程序
                LCALL   DELAY               ;延迟 39uS
                MOV     COM,#01H            ;清除屏幕显示，将 DDRAM 的地址计数器归零
                ACALL   WRITE_COMMAND       ;调用写指令子程序
                LCALL   DELAY               ;延迟 39uS
                MOV     COM,#06H            ;DDRAM 的地址计数器（AC）加 1
                ACALL   WRITE_COMMAND       ;调用写指令子程序
                LCALL   DELAY               ;延迟 39uS
                RET
;**********************液晶初始化模块子程序结束
;*****************字符显示子程序
PUT_STRING:
                MOV A,COM
                ORL A,#80H
                MOV COM,A
                LCALL  WRITE_COMMAND

DISP_STR_LOOP:
                MOV A,#0
```

```
                MOVC    A,@A+DPTR
                CJNE    A,#0,CONT_STR_DISP
                RET

CONT_STR_DISP:
                MOV DAT,A
                LCALL  WRITE_DATA
                INC DPTR
                SJMP    DISP_STR_LOOP

;*********************字符显示子程序结束
;************************写命令子程序
WRITE_COMMAND:
                PUSH     ACC
                CLR      RS
                SETB     RW
WRI1:    MOV       D_PORT,#0FFH
                SETB     E
                MOV      A,D_PORT
                CLR      E
                JB       ACC.7,WRI1
                CLR      RW
```

```
                MOV     D_PORT,COM
                SETB    E
                CLR     E
                POP     ACC
                RET

;********************写命令子程序结束
;************************写数据子程序
WRITE_DATA:
                PUSH    ACC
                CLR     RS
                SETB    RW
WRD1:   MOV     D_PORT,#0FFH
                SETB    E
                MOV     A,D_PORT
                CLR     E
                JB      ACC.7,WRD1
                SETB    RS
                CLR     RW
                MOV     D_PORT,DAT
                SETB    E
                CLR     E
```

```
                POP     ACC
                RET

;**********************写数据子程序结束
;**********************相关时间显示的子程序

BUSY:
                MOV R0,#0AH
                MOVX    A,@R0
                JB  ACC.7,BUSY
                RET

TDISP:

                MOV A,22H               ;22H 单元用来暂存产
生 bcd 码的数
                ANL A,#0F0H
                SWAP    A
                MOV DPTR,#SHU
                MOVC    A,@A+DPTR
                MOV DAT,A
                LCALL   WRITE_DATA
```

```
                MOV A,22H
                ANL A,#0FH
                MOV DPTR,#SHU
                MOVC    A,@A+DPTR
                MOV DAT,A
                LCALL   WRITE_DATA
                RET

HANZI:
                MOV 2DH,#00H            ;每个汉字查两次的
```

标志位

```
HANZI1:         MOV DPTR,#STRING2
                MOV A,TAB2
                CJNE    A,#0DH,TDISP1
                MOV TAB2,#00H

                AJMP    TDISP2
TDISP1:         INC TAB2

TDISP2:         MOVC    A,@A+DPTR
                MOV DAT,A
                LCALL   WRITE_DATA
```

```
                INC 2DH
                MOV A,2DH
                CJNE    A,#02H,HANZI1
                RET


;************************相关时间显示的子程序结束
;************************BCD 码子程序

BIN_BCD:
        MOV     2EH,R0
                LCALL   BUSY
                MOV R0,2EH
                MOVX    A,@R0

        MOV     B,#10
                DIV AB
                SWAP    A
                ORL     A,B
                MOV 22H,A
        RET
;************************BCD 码子程序结束
```

```
;*****************************相关的表目
STRING_ZHU:
        DB   "功 <1.测速模式> 菜 <3.其它功能> 能 <2.数据传输> 单 <4.退出系统> ",0
STRING_CESU:
                DB   0A1H,0F4H,"测速模式",0A1H,0F4H,"2.275 型  车型  1.26 型          3.返回    ",0
STRING_CHUANSHU:
                DB   0A1H,0F4H,"数据传输",0A1H,0F4H,"2.停止返回      1.开始传输  ",0
START_CHUAN:
                DB "正在传输....",0
FINISH_CHUAN:
                DB    "数据传输完毕!",0
STRING_FUNCTION:
                DB   0A1H,0F4H,"其它功能",0A1H,0F4H,"2.产品信息      1.时间设定      3.返回",0

STRING_SAMPLE:
        DB   0A1H,0F4H,"采样时间设定",0A1H,0F4H,"2.30 秒          1. 2 秒          3.60 秒",0
```

```
    STRING2:
                    DB  "年月日周时分秒"

    STRING_TIME:
            DB   0A1H,0F4H,"时间设定",0A1H,0F4H,0

    STRING_OK:
            DB    "OK",0
    SHU:
                    DB  "0123456789.m/skm:"

    HANG1:
                    DB  0A1H,0ECH,"速度:",0
    HANG2:
                    DB  0A1H,0ECH,"里程:",0
    HANG3:
                    DB  0A1H,0ECH,0A1H,0BCH,"      温      度
",0A1H,0BDH,0
    HANG4:
                    DB  0A1H,0ECH,0A1H,0BCH,0
    HANG5:
                    DB  0A1H,0E6H,0A1H,0BDH,0
```

```
START:
    DB
00H,00H,00H,00H,00H,00H,00H,00H,00H,00H,00H,00H,00H,00H
,00H
    DB  00H,00H,00H,00H,  00H,00H,00H,00H,  00H,00H,00H,00H,
00H,00H,00H
    DB
00H,00H,00H,00H,00H,20H,00H,00H,78H,00H,00H,00H,00H,00H,00H
,00H,00H
    DB  00H,00H,00H,03H,  3EH,00H,00H,86H,  00H,00H,00H,00H,
00H,00H,00H,00H
    DB  00H,00H,00H,02H,  0D0H,00H,00H,87H,  00H,00H,00H,00H,
00H,00H,00H,00H
    DB  00H,00H,02H,07H,  9EH,00H,01H,3FH,  7FH,0FFH,00H,00H,
00H,00H,00H,00H
    DB 00H,20H,73H,0C9H, 0D0H,00H,07H,7FH, 00H,03H,0F0H,00H,
00H,00H,00H,00H
    DB  00H,61H,0EFH,42H,  5EH,00H,00H,7EH,  00H,00H,1FH,00H,
00H,00H,00H,00H
    DB          03H,0E1H,0CAH,0C3H,          0D0H,80H,00H,7EH,
00H,00H,01H,0E0H, 00H,00H,00H,00H
    DB  0DH,20H,52H,82H,  50H,80H,00H,1EH,  00H,00H,00H,3CH,
```

00H,00H,00H,00H

DB 31H,0E0H,72H,83H, 5FH,00H,00H,06H, 00H,00H,00H,7EH,
00H,00H,00H,00H

DB 06H,30H,0E6H,82H, 40H,00H,00H,02H, 00H,00H,00H,0FFH,
80H,00H,00H,00H

DB 39H,0C8H,8CH,86H, 00H,00H,00H,03H, 00H,00H,00H,0FFH,
80H,00H,00H,00H

DB 06H,0B0H,1CH,80H, 00H,00H,00H,01H, 00H,00H,03H,0FFH,
80H,00H,00H,00H

DB 18H,0C0H,38H,00H, 00H,00H,00H,00H, 83H,0E0H,0FH,0FFH,
0C0H,00H,00H,00H

DB 01H,80H,00H,00H, 00H,00H,00H,00H, 0C3H,0FEH,1FH,0FFH,
0C0H,00H,00H,00H

DB 06H,00H,00H,00H, 00H,00H,00H,00H, 6FH,0FFH,0BFH,0FFH,
0C0H,00H,00H,00H

DB 3CH,00H,00H,00H, 00H,00H,00H,00H, 3FH,1FH,0FFH,0FFH,
0C0H,00H,00H,00H

DB 18H,00H,00H,00H, 00H,00H,00H,00H, 3FH,03H,0FFH,0FFH,
80H,00H,00H,00H

DB 00H,00H,00H,00H, 00H,00H,00H,00H, 1FH,01H,0FFH,0FFH,
00H,00H,00H,00H

DB 00H,00H,00H,00H, 00H,00H,00H,00H, 1FH,00H,0FFH,0FFH,

00H,00H,00H,00H

DB 00H,00H,00H,00H, 00H,00H,00H,00H, 1FH,00H,0FFH,0FEH,
00H,00H,00H,00H

DB 00H,00H,00H,00H, 00H,00H,00H,00H, 1FH,00H,0FFH,0FCH,
00H,00H,00H,00H

DB 00H,00H,00H,00H, 00H,00H,00H,00H, 1FH,00H,0FFH,0F8H,
00H,00H,00H,00H

DB 00H,00H,00H,00H, 00H,00H,00H,00H, 1FH,00H,0FFH,0F0H,
00H,00H,00H,00H

DB 00H,00H,00H,00H, 00H,00H,00H,00H, 1EH,00H,0FFH,0F0H,
00H,00H,00H,00H

DB 00H,00H,00H,00H, 00H,00H,00H,00H, 1EH,00H,0FFH,0E0H,
00H,00H,00H,00H

DB 01H,00H,40H,00H, 80H,00H,00H,00H, 1CH,00H,0FFH,0E7H,
70H,00H,00H,00H

DB 00H,80H,4FH,80H, 80H,00H,00H,00H, 7FH,0C0H,0FFH,0FFH,
0F8H,00H,00H,00H

DB 07H,0F8H,80H,0FH, 0FCH,00H,00H,00H,
0FFH,0C0H,0FFH,0C7H, 80H,00H,00H,00H

DB 04H,08H,0A0H,01H, 00H,00H,00H,00H, 18H,00H,0FFH,0C7H,
0C0H,00H,00H,00H

DB 04H,08H,2FH,0C2H, 40H,00H,00H,00H, 1CH,00H,0FFH,0C6H,

0E0H,00H,00H,00H

DB 07H,0F8H,41H,04H, 40H,00H,00H,00H, 37H,00H,0FFH,0CCH,
70H,00H,00H,00H

DB 04H,08H,0C1H,07H, 0F8H,00H,00H,00H,
31H,0C0H,0FFH,0CCH, 38H,00H,00H,00H

DB 07H,0F8H,41H,00H, 40H,00H,00H,00H, 60H,0E0H,7FH,0CCH,
1CH,00H,00H,00H

DB 04H,08H,41H,0FH, 0FCH,00H,00H,00H, 60H,78H,7FH,0D8H,
0EH,1FH,0C0H,00H

DB 07H,0F8H,41H,00H, 40H,00H,00H,00H, 60H,1CH,3FH,0D8H,
07H,0FFH,0F8H,00H

DB 04H,08H,43H,00H, 40H,00H,00H,7FH, 0F0H,07H,3FH,0D8H,
07H,0FFH,0FEH,00H

DB 00H,00H,00H,00H, 00H,00H,03H,0FFH, 0FEH,01H,0DFH,0F0H,
1FH,0FFH,0FFH,80H

DB 00H,00H,00H,00H, 00H,00H,0FH,0FFH, 0FFH,80H,0FFH,0F0H,
3FH,0FFH,0F9H,0C0H

DB 00H,00H,00H,00H, 00H,00H,1FH,7FH, 0FFH,0C0H,7FH,0F0H,
7FH,0FFH,0F0H,0E0H

DB 00H,00H,00H,00H, 00H,00H,3CH,7FH, 0FFH,0E0H,1FH,0F0H,
0FFH,0FFH,0E0H,0F0H

DB 00H,00H,00H,00H, 00H,00H,78H,3FH, 0FFH,0F0H,0FH,0F0H,

0FFH,0FFH,0C3H,0F8H

DB 00H,00H,00H,00H, 00H,00H,0FEH,1FH, 0FFH,0F8H,07H,0F1H,
0FFH,0FFH,8FH,0F8H

DB 00H,00H,00H,00H, 00H,00H,0FFH,1FH, 0FFH,0FCH,03H,0F3H,
0FFH,0FFH,0BFH,0FCH

DB 00H,00H,02H,00H, 00H,01H,0FFH,0CFH,
0FFH,0FCH,03H,0F3H, 0FFH,0FFH,0FFH,0FCH

DB 0BH,0C4H,9FH,0C2H, 40H,01H,0FFH,0FFH,
0FFH,0FCH,03H,0FBH, 0FFH,0FFH,0FFH,0FCH

DB 06H,54H,42H,04H, 28H,01H,0FFH,0FFH,
0FFH,0FCH,01H,0FBH, 0FFH,0FFH,0FFH,0FCH

DB 02H,0D4H,5FH,0C4H, 88H,01H,0FFH,0FFH,
0FFH,0FCH,01H,0FBH, 0FFH,0FFH,0FFH,0FCH

DB 0AH,0D4H,12H,4CH, 90H,01H,0FFH,0FFH,
0FFH,0FCH,00H,0F3H, 0FFH,0FFH,0FFH,0FCH

DB 06H,0D4H,0DFH,54H, 90H,01H,0FFH,0FFH,
0FFH,0FCH,00H,0E1H, 0FFH,0FFH,0FFH,0F8H

DB 02H,0D4H,46H,04H, 50H,01H,0FFH,0FFH,
0FFH,0FCH,00H,81H, 0FFH,0FFH,0FFH,0F0H

DB 06H,0D4H,4AH,84H, 60H,00H,0FFH,0FFH,
0FFH,0FCH,00H,00H, 0FFH,0FFH,0FFH,0F0H

DB 04H,0D4H,52H,44H, 60H,00H,0FFH,0FFH,

0FFH,0F8H,00H,00H, 7FH,0FFH,0FFH,0E0H

DB 09H,44H,42H,04H, 90H,00H,7FH,0FFH, 0FFH,0F0H,00H,00H, 3FH,0FFH,0FFH,0C0H

DB 0AH,2CH,0BFH,0C5H, 0CH,00H,3FH,0FFH, 0FFH,0E0H,00H,00H, 1FH,0FFH,0FFH,80H

DB 00H,00H,00H,00H, 00H,00H,1FH,0FFH, 0FFH,0C0H,00H,00H, 0FH,0FFH,0FEH,00H

DB 00H,00H,00H,00H, 00H,00H,0FH,0FFH, 0FFH,80H,00H,00H, 01H,0FFH,0F8H,00H

DB 00H,00H,00H,00H, 00H,07H,0FFH,0FFH, 0FFH,0FFH,0FFH,0FFH, 0FFH,0FFH,0FFH,0FFH

DB 00H,00H,00H,00H, 00H,07H,0FFH,0FFH, 0FFH,0FFH,0FFH,0FFH, 0FFH,0FFH,0FFH,0FFH

DB 00H,00H,00H,00H, 00H,07H,0FFH,0FFH, 0FFH,0FFH,0FFH,0FFH, 0FFH,0FFH,0FFH,0FFH

DB 00H,00H,00H,00H, 00H,07H,0FFH,0FFH, 0FFH,0FFH,0FFH,0FFH, 0FFH,0FFH,0FFH,0FFH

DB

0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH ,0FFH,0FFH,0FFH,0FFH

DB

0C0H,00H,00H,00H,00H,00H,00H,00H,00H,00H,00H,00H,00H,00

```
H,03H


    FINISH:
    DB  00H,00H,00H,01H,  00H,00H,00H,00H,  00H,00H,00H,00H,
00H,01H,10H,00H
    DB  00H,00H,00H,01H,  00H,00H,00H,00H,  00H,00H,00H,00H,
00H,7FH,0FCH,00H
    DB  00H,00H,00H,01H,  80H,00H,00H,00H,  00H,00H,00H,00H,
00H,40H,80H,00H
    DB  00H,00H,00H,01H,  80H,00H,00H,00H,  00H,00H,00H,00H,
00H,5FH,50H,00H
    DB  00H,00H,00H,00H,  80H,00H,00H,00H,  00H,00H,00H,00H,
00H,40H,20H,00H
    DB  00H,00H,00H,00H,  0C0H,00H,00H,00H,  00H,00H,00H,00H,
00H,5FH,0D0H,00H
    DB  00H,00H,00H,00H,  60H,00H,00H,00H,  00H,00H,00H,00H,
00H,0D1H,58H,00H
    DB  00H,00H,00H,00H,  38H,00H,00H,00H,  00H,00H,00H,00H,
00H,3EH,0CH,00H
    DB  00H,00H,00H,00H,  1FH,00H,00H,00H,  00H,00H,00H,00H,
00H,25H,90H,00H
    DB  00H,00H,00H,00H,  03H,0E0H,00H,00H,  00H,00H,00H,00H,
```

00H,0C3H,01H,00H

DB 00H,00H,00H,00H, 00H,78H,00H,00H, 00H,00H,00H,00H,
00H,81H,0C2H,00H

DB 00H,00H,00H,00H, 00H,06H,00H,00H, 00H,00H,00H,00H,
00H,00H,7EH,00H

DB 00H,00H,00H,00H, 00H,03H,80H,00H, 00H,00H,00H,00H,
00H,00H,08H,00H

DB 00H,00H,00H,00H, 00H,00H,0C0H,00H, 00H,00H,00H,00H,
00H,00H,00H,00H

DB 00H,00H,00H,00H, 00H,00H,60H,00H, 00H,00H,00H,00H,
00H,00H,00H,00H

DB 00H,00H,00H,00H, 00H,00H,30H,00H, 00H,00H,00H,00H,
00H,00H,00H,00H

DB 00H,00H,00H,00H, 00H,00H,7FH,80H, 00H,00H,00H,00H,
00H,00H,00H,00H

DB 00H,00H,00H,00H, 00H,01H,0FFH,0C0H, 08H,00H,00H,00H,
00H,00H,00H,00H

DB 00H,00H,00H,00H, 00H,01H,0FFH,0E0H, 10H,00H,00H,00H,
00H,0F1H,04H,00H

DB 00H,00H,00H,00H, 00H,03H,0FFH,0F0H, 10H,00H,00H,00H,
00H,0FBH,0E4H,00H

DB 00H,00H,00H,00H, 00H,03H,0FFH,0F0H, 20H,00H,00H,00H,

00H,0CH,44H,00H

DB 00H,00H,00H,00H, 00H,03H,0FFH,0F8H, 00H,00H,00H,00H,
00H,07H,0FFH,00H

DB 00H,00H,00H,00H, 00H,01H,0FFH,0FFH, 80H,06H,00H,00H,
00H,0E4H,7FH,00H

DB 00H,00H,00H,00H, 00H,01H,0FFH,0FFH, 80H,78H,00H,00H,
00H,2FH,04H,00H

DB 00H,00H,00H,00H, 00H,00H,0FFH,0E7H, 81H,80H,00H,00H,
00H,24H,0B4H,00H

DB 00H,00H,00H,00H, 00H,00H,0FFH,0E7H, 0C0H,00H,00H,00H,
00H,31H,0A4H,00H

DB 00H,00H,00H,00H, 00H,00H,79H,0F3H, 80H,00H,00H,00H,
00H,32H,84H,00H

DB 00H,00H,00H,00H, 00H,00H,0F8H,0FEH, 00H,00H,00H,00H,
00H,6CH,8CH,00H

DB 00H,00H,00H,00H, 00H,01H,0FCH,0F8H, 00H,00H,00H,00H,
00H,0C0H,0BCH,00H

DB 00H,00H,00H,00H, 01H,0F0H,0FDH,0C0H, 00H,00H,00H,00H,
00H,00H,00H,00H

DB 00H,00H,00H,00H, 00H,00H,7EH,03H, 00H,1CH,00H,00H,
00H,00H,00H,00H

DB 00H,00H,00H,00H, 00H,00H,30H,03H, 84H,03H,80H,00H,

```
00H,00H,00H,00H

    DB  00H,00H,00H,00H,  00H,00H,00H,01H,  0CFH,00H,00H,00H,
00H,00H,00H,00H

    DB 00H,00H,00H,00H,  00H,00H,00H,00H,  0FFH,0C0H,00H,00H,
00H,00H,00H,00H

    DB  00H,00H,00H,00H,  00H,00H,00H,30H,  7FH,0E0H,00H,00H,
00H,00H,00H,00H

    DB 00H,00H,00H,00H,  00H,04H,00H,18H,  0FFH,0C0H,00H,00H,
00H,00H,40H,00H

    DB 00H,00H,00H,00H,  00H,08H,00H,0DH,  0FFH,0E0H,00H,00H,
00H,1FH,0FFH,00H

    DB
00H,00H,00H,00H,00H,10H,00H,07H,0FFH,0F0H,00H,00H,00H,0F0H,
20H,00H

    DB
00H,00H,00H,00H,00H,60H,00H,07H,0FFH,0F8H,00H,00H,00H,27H,0
FEH,00H,00H

    DB
00H,00H,00H,00H,00H,10H,0FH,0FFH,0FCH,00H,00H,00H,22H,24H,0
0H,00H

    DB
00H,00H,00H,00H,00H,10H,3FH,0FFH,0FEH,00H,00H,00H,27H,0FCH,
```

```
00H,00H
    DB
00H,00H,00H,00H,00H,10H,1FH,0FFH,0FFH,00H,00H,00H,21H,40H,0
0H,00H
    DB
00H,00H,00H,00H,00H,20H,0FH,0FFH,0FFH,00H,00H,00H,20H,0C0H,
00H,00H
    DB
00H,00H,00H,00H,00H,20H,06H,0FFH,0FEH,00H,00H,00H,20H,0C0H,
00H,00H,00H
    DB
00H,00H,00H,00H,20H,00H,7FH,0FFH,80H,00H,00H,2FH,0B0H,00H,0
0H,00H
    DB
00H,00H,00H,00H,00H,00H,3FH,0F8H,0E0H,00H,00H,2EH,1FH,00H,0
0H,00H
    DB
00H,00H,00H,00H,00H,00H,0FH,0F0H,38H,00H,00H,00H,06H,00H,00
H,00H,00H,00H
    DB
00H,00H,00H,00H,03H,0C0H,0CH,00H,00H,00H,00H,00H,00H,00H,00
H,03H
```

```
    DB
00H,00H,00H,00H,00H,00H,06H,00H,00H,00H,00H,00H,00H,60H,00H
,04H,38H,00H
    DB
00H,00H,00H,00H,03H,00H,00H,00H,00H,00H,01H,98H,00H,04H
    DB
44H,00H,18H,00H,00H,00H,01H,00H,00H,00H,00H,00H,02H,08H,00H
,09H
    DB
0B4H,00H,10H,00H,00H,00H,01H,80H,00H,00H,00H,00H,06H,01H,86
H,39H
    DB
25H,93H,90H,00H,00H,00H,00H,0C0H,00H,00H,00H,00H,04H,02H,49
H,48H
    DB
38H,94H,90H,00H,00H,00H,00H,60H,00H,3FH,0FEH,00H,04H,04H,51
H,48H
    DB
24H,97H,90H,00H,00H,00H,00H,30H,00H,3FH,0FEH,00H,04H,74H,51
H,90H
    DB
44H,5CH,00H,00H,00H,00H,00H,1CH,00H,21H,84H,00H,04H,24H,92H
```

,0B4H

DB

44H,69H,00H,00H,00H,00H,00H,07H,00H,3FH,0FCH,00H,03H,0E3H,0CH

DB

0D8H,0C8H,6EH,20H,00H,00H,00H,00H,01H,0E0H,39H,84H,00H,00H,40H,00H

DB

00H,30H,40H,00H,00H,00H,00H,00H,00H,00H,21H,84H,00H,00H,40H,00H,00H

DB

00H,80H,00H,00H,00H,00H,00H,00H,00H,3FH,0FCH,00H,02H,80H,00H,00H,01H

DB

80H,00H,00H,00H,00H,00H,00H,00H,61H,84H,00H,03H,00H,00H,00H,02H,00H

DB

00H,00H,00H,00H,00H,00H,01H,0E1H,84H,00H,00H,00H,00H,00H,00H,00H,00H

DB

00H,00H,00H,00H,00H,01H,0C1H,04H,00H,00H,00H,00H,00H,00H,00H,00H,00H

```
DB 00H,00H,00H,00H,00H,81H,04H,00H

END
```