 **单片机实验报告**

**系 别 电气信息系**

**专 业 计算机科学与技术**

**班 级 17gb 计算3班**

**姓 名 唐可寅**

**学 号 201710253110**

**实验一 LED实验**

**一、实验目的**

**1、了解I/O访问方式**

**2、熟悉简单的程序设计**

**二、实验设备**

计算机，STC单片机下载线，单片机教学系统。

**三、实验原理**

1、LED，英文全称是Light Emitting Diode，翻译成中文是发光二极管，最常用的电子器件之一。

2、本实验是由单片机通过I/O接口对发光二极管进行控制，由数据总线P2口的八位控制8个LED发光二极管。其中8个LED发光二极管为LED1-LED8。

**四、实验步骤**

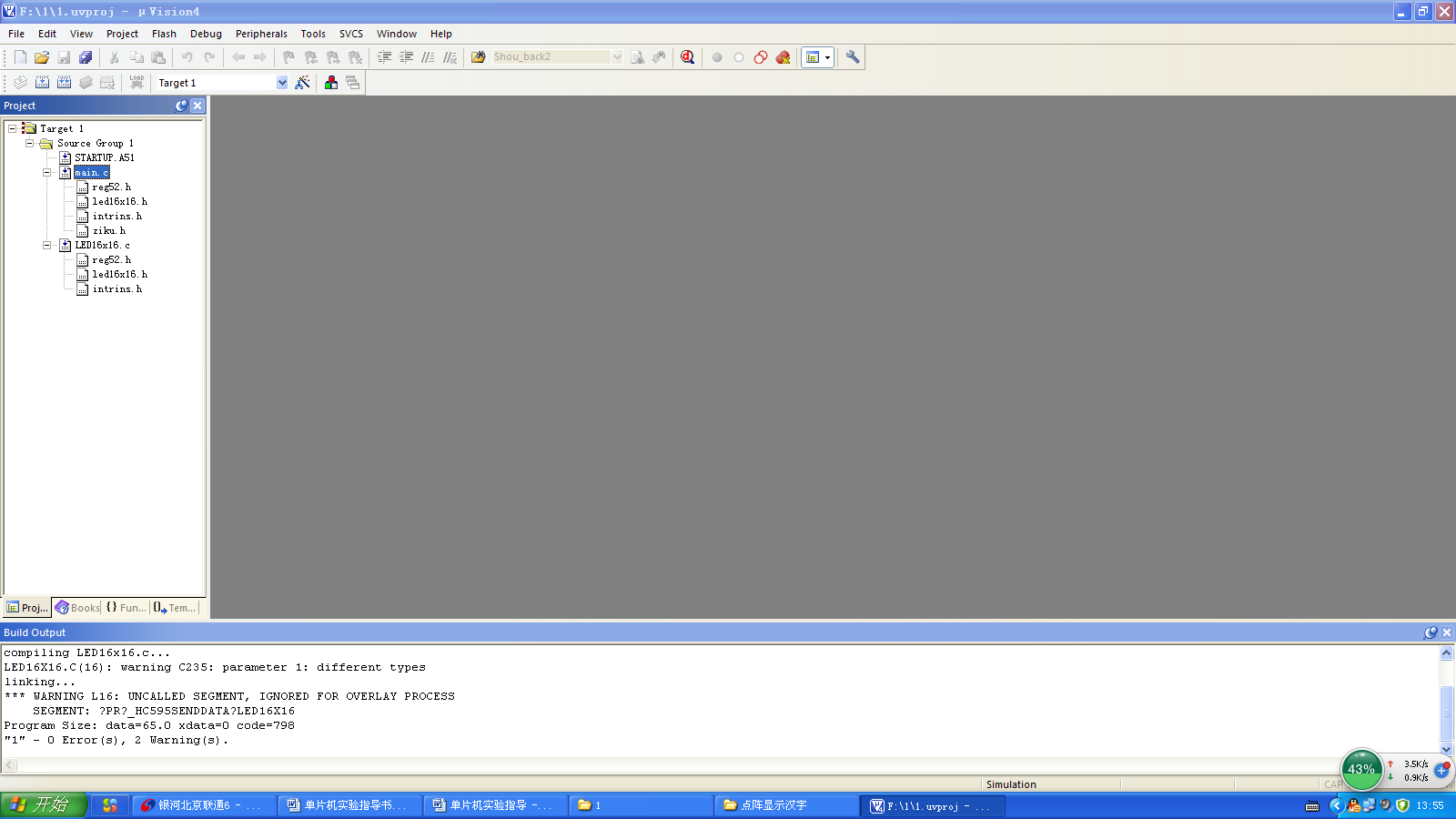
1、在可写入的F盘，新建一个工作文件夹，例如：文件夹名为“1”；

2、双击“”，进入“Keil uVision4”软件环境；

3、选择菜单栏中的“Project”项目中“New uVision Project…”选项，命名建立一个\*.pjt工程项目，弹出“Select a CPU Data Base File”窗口中，选择目标芯片系列：Generic CPU Data Base，点击“OK”；在弹出窗口“Data Base”项目中，选择“Atmel”，再选择“AT89C51”芯片，点击“OK”，弹出“Copy ‘STARTUP.A51’to Project Folder and Add File to Project”窗口中，选择“是”，项目建立完成。

4、将主窗口左侧工程管理窗口中“Target 1”的“+”点开，选择菜单栏中的“File”项目中“New”，根据类型输入源程序后另存为\*.h或 \*.c到\*.pjt所在的文件夹根目录下，注意后缀名.h及.c需要手动输入。项目中所需文件建立完成。

5、选择主窗口左侧工程管理窗口中“Target 1”的“+”点开，鼠标右键“Source Group”，选择菜单中“Add Files to Group ‘Soucre Group 1’ …”，弹出窗口中选择刚才保存的\*.c文件（注意\*.h不能添加），点击“Add”，然后点击“Close”，关闭该窗口。项目组建完成。

6、点击快捷工具栏中“”（Rebuild），对照报错窗口中的提示改错，当报错窗口中的错误（Errors）、警告（Warnings）均无误后，出现提示信息：0 Error（s）, 0 Warning（s），项目生成结果文件\*.hex（注意：点击“Project”中“Options for Target”，出现窗口点击“output”，勾选“Create HEX File”）。

7、下载程序结果，运用STC-ISP软件独立下载\*.hex结果文件到单片机教学系统。

8、运行程序结果，在教学系统对应的显示模块中出现相应的现象。

**五、写出实验源程序**

**//LED.c**

**#include<reg51.h>**

**#include <intrins.h>**

**void delayms(unsigned char ms)**

**{**

**unsigned char i;**

**while(ms--)**

**{**

**for(i = 0; i < 120; i++);**

**}**

**}**

**void main()**

**{**

**unsigned char LED;**

**LED = 0xfe;**

**P2 = LED;**

**while(1)**

**{**

**delayms(250);**

**LED = \_crol\_(LED,1); //循环右移1位，点亮下一个LED**

**P2 = LED;**

**}**

**}**

**六、针对本实验写出实验心得**

**通过本次实验了解了单片机和LED的工作原理、I/O访问方式、单片机硬件制式，熟悉了Keil uVision4和STC-ISP软件的使用、程序编译和烧录方式。在烧录时应注意串口连线和对串口的保护，避免错误操作导致串口烧毁。**

**实验二 数码管实验**

**一、实验目的**

**1、了解数码管的接口定义及使用**

**2、熟悉单片机控制数码管的程序设计**

**二、实验设备**

计算机，STC单片机下载线，单片机教学系统。

**三、实验原理**

1、[led](http://baike.baidu.com/view/52538.htm" \t "_blank)数码管（LED Segment Displays）由多个[发光二极管](http://baike.baidu.com/view/84213.htm" \t "_blank)封装在一起组成“8”字型的器件，引线已在内部连接完成，只需引出它们的各个笔划，公共电极。[数码管](http://baike.baidu.com/view/556862.htm" \t "_blank)实际上是由七个发光管组成8字形构成的，加上小数点就是8个。这些段分别由字母a,b,c,d,e,f,g,dp来表示。

2、本实验是由单片机通过I/O接口对数码管进行控制，由数据总线P2口控制数码管显示，数据总线P1口接收矩阵键盘信息配合显示。

**四、实验步骤**

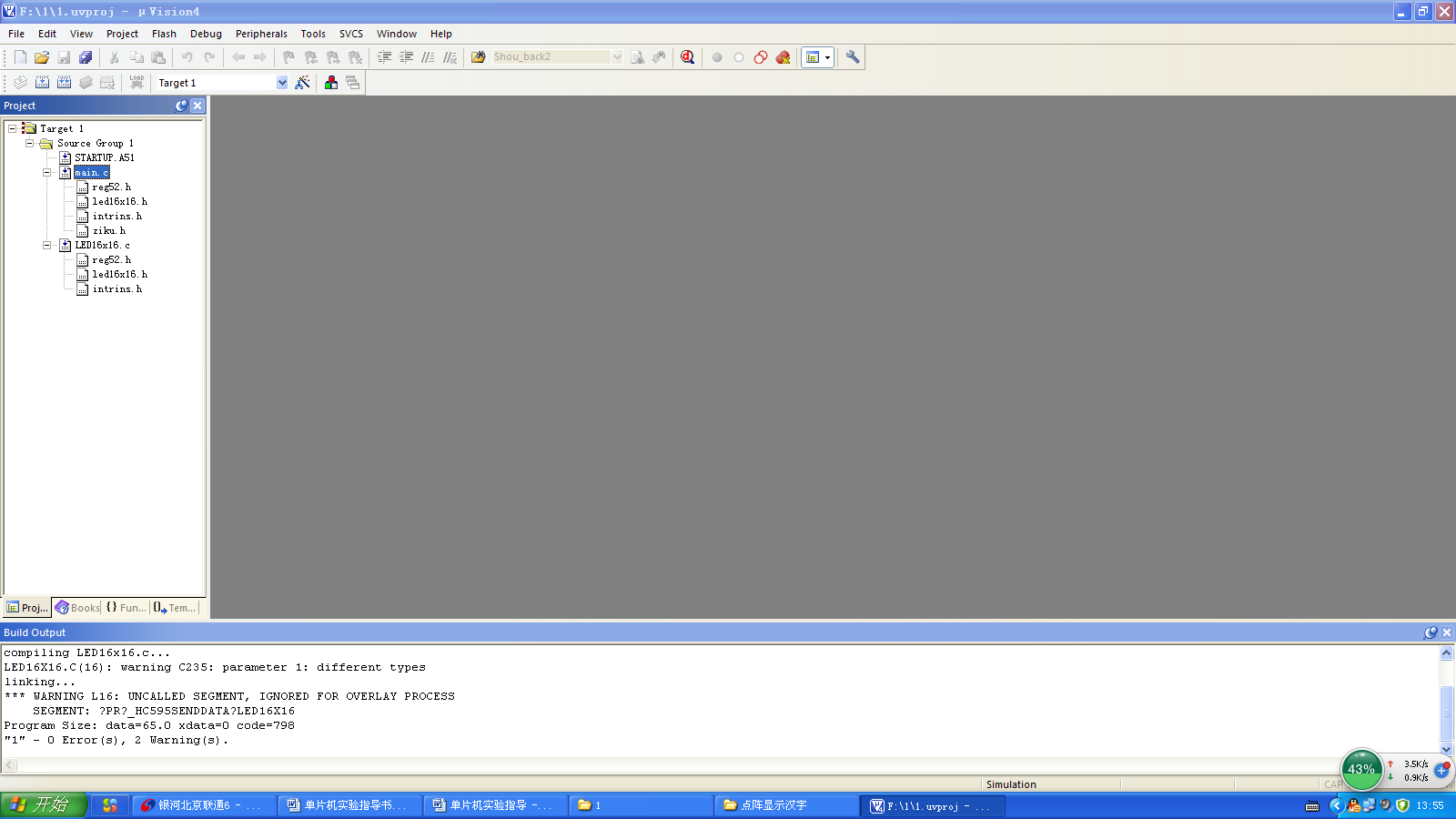
1、在可写入的F盘，新建一个工作文件夹，例如：文件夹名为“1”；

2、双击“”，进入“Keil uVision4”软件环境；

3、选择菜单栏中的“Project”项目中“New uVision Project…”选项，命名建立一个\*.pjt工程项目，弹出“Select a CPU Data Base File”窗口中，选择目标芯片系列：Generic CPU Data Base，点击“OK”；在弹出窗口“Data Base”项目中，选择“Atmel”，再选择“AT89C51”芯片，点击“OK”，弹出“Copy ‘STARTUP.A51’to Project Folder and Add File to Project”窗口中，选择“是”，项目建立完成。

4、将主窗口左侧工程管理窗口中“Target 1”的“+”点开，选择菜单栏中的“File”项目中“New”，根据类型输入源程序后另存为\*.h或 \*.c到\*.pjt所在的文件夹根目录下，注意后缀名.h及.c需要手动输入。项目中所需文件建立完成。

5、选择主窗口左侧工程管理窗口中“Target 1”的“+”点开，鼠标右键“Source Group”，选择菜单中“Add Files to Group ‘Soucre Group 1’ …”，弹出窗口中选择刚才保存的\*.c文件（注意\*.h不能添加），点击“Add”，然后点击“Close”，关闭该窗口。项目组建完成。

6、点击快捷工具栏中“”（Rebuild），对照报错窗口中的提示改错，当报错窗口中的错误（Errors）、警告（Warnings）均无误后，出现提示信息：0 Error（s）, 0 Warning（s），项目生成结果文件\*.hex（注意：点击“Project”中“Options for Target”，出现窗口点击“output”，勾选“Create HEX File”）。

7、下载程序结果，运用STC-ISP软件独立下载\*.hex结果文件到单片机教学系统。

8、运行程序结果，在教学系统对应的显示模块中出现相应的现象。

**五、写出实验源程序**

**//SMG.c**

**#include <intrins.h>**

**#include<reg52.h>**

**#define AT24C02 0xa0 //AT24C02 地址**

**#define NOP() \_nop\_() /\* 定义空指令 \*/**

**sbit MOSIO =P2^0;**

**sbit R\_CLK =P2^1;**

**sbit S\_CLK =P2^2;**

**void delay(unsigned int i); //函数声名**

**void HC595SendData(unsigned char SendVal,unsigned char Wei);**

**void Led\_Show(unsigned char Wei);**

**void SetLedNum(unsigned long int Numcode);**

**void system\_Ini();**

**void keyscan(void);**

**void SengUart(unsigned char SenData);**

**void SendString(unsigned char \*str);**

**void SengNum(unsigned int num);**

**unsigned char code Disp\_Tab[] = { 0xC0,0xF9,0xA4,0xB0,0x99,0x92,0x82,0xF8,0x80,0x90,0x88,0x83,0xC6,0xA1,0x86,0xbf,0xc7,0x8c,0xc1, 0xff, 0xf7 };**

**unsigned char code LED7Code[] = {~0x3F,~0x06,~0x5B,~0x4F,~0x66,~0x6D,~0x7D,~0x07,~0x7F,~0x6F,~0x77,~0x7C,~0x39,~0x5E,~0x79,~0x71};**

**unsigned char code Nuntable[]="0123456789abcdef";**

**unsigned char NumBuffer[8];**

**unsigned int LedNum=0;**

**unsigned int time=0;**

**unsigned char P0flg;**

**unsigned char temp;**

**unsigned char key; //键顺序吗**

**unsigned char ReData;**

**unsigned char pDat[8];**

**main()**

**{**

**unsigned long int Num=0;**

**P0=0xff;**

**P1=0xff;**

**P2=0xff;**

**system\_Ini();**

**P0flg=0;**

**NumBuffer[6]=pDat[5];**

**while(1)**

**{**

**if(LedNum==0)**

**{**

**Num++;**

**Num%=10000;**

**SetLedNum(Num);**

**}**

**keyscan();**

**}**

**}**

**void system\_Ini()**

**{**

**TMOD= 0x21;**

**TH0 = (65536-30000)>>8; //12.000**

**TL0 = (65536-30000)&0xff;**

**ET0 =1;**

**TR0 = 1;**

**SCON = 0x50; //REN=1允许串行接受状态，串口工作模式2**

**TMOD|= 0x20; //定时器工作方式2**

**PCON|= 0x80; //波特率提高一倍**

**TH1 = 0xF3;// //baud\*2 /\* 波特率4800、数据位8、停止位1。**

**TL1 = 0xF3;**

**TR1 = 1; //开启定时器1**

**ES = 1; //开串口中断**

**IT0=1; //下降沿触发**

**EX0=1;**

**EA = 1; // 开总中断**

**}**

**void SetLedNum(unsigned long int Numcode)**

**{**

**unsigned char i;**

**for(i=0;i<6;i++)**

**{**

**NumBuffer[i]=Numcode%10;**

**Numcode/=10;**

**}**

**}**

**void Led\_Show(unsigned char Wei)**

**{**

**unsigned char HC595SendVal;**

**HC595SendVal = ~Disp\_Tab[NumBuffer[Wei]];**

**HC595SendData(HC595SendVal,Wei);**

**}**

**void delay(unsigned int i)**

**{**

**unsigned int j;**

**for(i; i > 0; i--)**

**for(j = 300; j > 0; j--);**

**}**

**void HC595SendData(unsigned char SendVal,unsigned char Wei)**

**{**

**unsigned char i;**

**for(i=0;i<16;i++)**

**{**

**if(i<8)**

**{**

**if((SendVal<<i)&0x80) MOSIO=1;**

**else MOSIO=0; // 如果为真 MOSIO = 1**

**}**

**else**

**{**

**MOSIO=((~(1<<Wei)>>(i-8))&0x01);**

**}**

**S\_CLK=0;**

**NOP();**

**NOP();**

**S\_CLK=1;**

**}**

**R\_CLK=0; //set dataline low**

**NOP();**

**NOP();**

**R\_CLK=1; //片选**

**}**

**void SengUart(unsigned char SenData)**

**{**

**SBUF=SenData; //SUBF接受/发送缓冲器**

**while(TI==0);**

**TI=0;**

**}**

**void SendString(unsigned char \*str)**

**{**

**while(\*str!='\0')**

**{**

**SengUart(\*str);**

**str++;**

**}**

**}**

**void SengNum(unsigned int num)**

**{**

**unsigned char buffer[10];**

**unsigned char \*Buf=buffer+8;**

**do{**

**\*Buf=Nuntable[num%10];**

**Buf--;**

**num/=10;**

**}while(num!=0);**

**buffer[9]=0;**

**Buf++;**

**SendString(Buf);**

**}**

**void keyscan(void)**

**{ temp = 0;**

**P1=0xF0; //高四位输入 行为高电平 列为低电平**

**delay(1);**

**temp=P1; //读P1口**

**temp=temp&0xF0; //屏蔽低四位**

**temp=~((temp>>4)|0xF0);**

**if(temp==1) // p1.4 被拉低**

**key=0;**

**else if(temp==2) // p1.5 被拉低**

**key=1;**

**else if(temp==4) // p1.6 被拉低**

**key=2;**

**else if(temp==8) // p1.7 被拉低**

**key=3;**

**else**

**key=16;**

**P1=0x0F; //低四位输入 列为高电平 行为低电平**

**delay(1);**

**temp=P1; //读P1口**

**temp=temp&0x0F;**

**temp=~(temp|0xF0);**

**if(temp==1) // p1.0 被拉低**

**key=key+0;**

**else if(temp==2) // p1.1 被拉低**

**key=key+4;**

**else if(temp==4) // p1.2 被拉低**

**key=key+8;**

**else if(temp==8) // p1.3 被拉低**

**key=key+12;**

**else**

**key=16;**

**if(key<16)**

**{**

**NumBuffer[7]=key;**

**SendString("get the key number: ");**

**SengNum((unsigned int)key);**

**SendString("\r\n");**

**if(key==0)**

**{**

**SendString("write num 5 to 24c02 !\r\n");**

**}**

**else if(key==1)**

**{**

**SendString("read num from 24c02 : ");**

**NumBuffer[6]=pDat[0];**

**SengNum((unsigned int)pDat[0]);**

**SendString("\r\n");**

**}**

**}**

**}**

**void counter(void) interrupt 0**

**{**

**EX0=0;**

**EX0=1;**

**}**

**void T1zd(void) interrupt 1 //3定时器0的中断号 1定时器0的中断号 0外部中断1 2外部中断2 4串口中断**

**{**

**TH0 = (65536-3000)>>8; //12.000**

**TL0 = (65536-3000)&0xff;**

**time++;**

**if(time==10)**

**{**

**P0flg++;**

**P0flg%=16;**

**time = 0;**

**if(P0flg<8)**

**P0=~(0x01<<P0flg);**

**else**

**P0=~(0x80>>(P0flg-8));**

**}**

**LedNum++; //中断计数**

**LedNum%=8;**

**Led\_Show(LedNum);**

**}**

**void ser\_int (void) interrupt 4 using 1**

**{**

**if(RI == 1) //RI接受中断标志**

**{**

**RI = 0; //清除RI接受中断标志**

**ReData = SBUF; //SUBF接受/发送缓冲器**

**}**

**}**

**六、针对本实验写出实验心得**

**通过本次实验了解了单片机和LED的工作原理、I/O访问方式、单片机硬件制式，熟悉了Keil uVision4和STC-ISP软件的使用、程序编译和烧录方式。在烧录时应注意串口连线和对串口的保护，避免错误操作导致串口烧毁。**

**实验三 点阵实验**

**一、实验目的**

**1、了解点阵的接口定义及使用**

**2、熟悉单片机控制点阵的程序设计**

**二、实验设备**

计算机，STC单片机下载线，单片机教学系统。

**三、实验原理**

1、LED点阵屏通过[LED](http://baike.baidu.com/view/52538.htm" \t "_blank)([发光二极管](http://baike.baidu.com/subview/84213/84213.htm" \t "_blank)）组成，以灯珠亮灭来显示文字、图片、动画、视频等，是各部分组件都模块化的[显示器件](http://baike.baidu.com/view/3844065.htm" \t "_blank)，通常由显示模块、[控制系统](http://baike.baidu.com/view/57978.htm" \t "_blank)及[电源系统](http://baike.baidu.com/view/3317818.htm" \t "_blank)组成。

2、8\*8[点阵](http://baike.baidu.com/view/545607.htm" \t "_blank)，它共由64个[发光二极管](http://baike.baidu.com/subview/84213/84213.htm" \t "_blank)组成，且每个发光二极管是放置在行线和列线的交叉点上，当对应的某一行置1[电平](http://baike.baidu.com/view/314180.htm" \t "_blank)，某一列置0电平，则相应的二极管就亮。一般我们使用点阵显示汉字是用的16\*16的点阵宋体[字库](http://baike.baidu.com/view/1127103.htm" \t "_blank)，所谓16\*16，是每一个汉字在纵、横各16点的区域内显示的。也就是说用四个8\*8点阵组合成一个16\*16的点阵。

3、本实验是由单片机通过I/O接口对16\*16点阵进行控制，显示汉字。

**四、实验步骤**

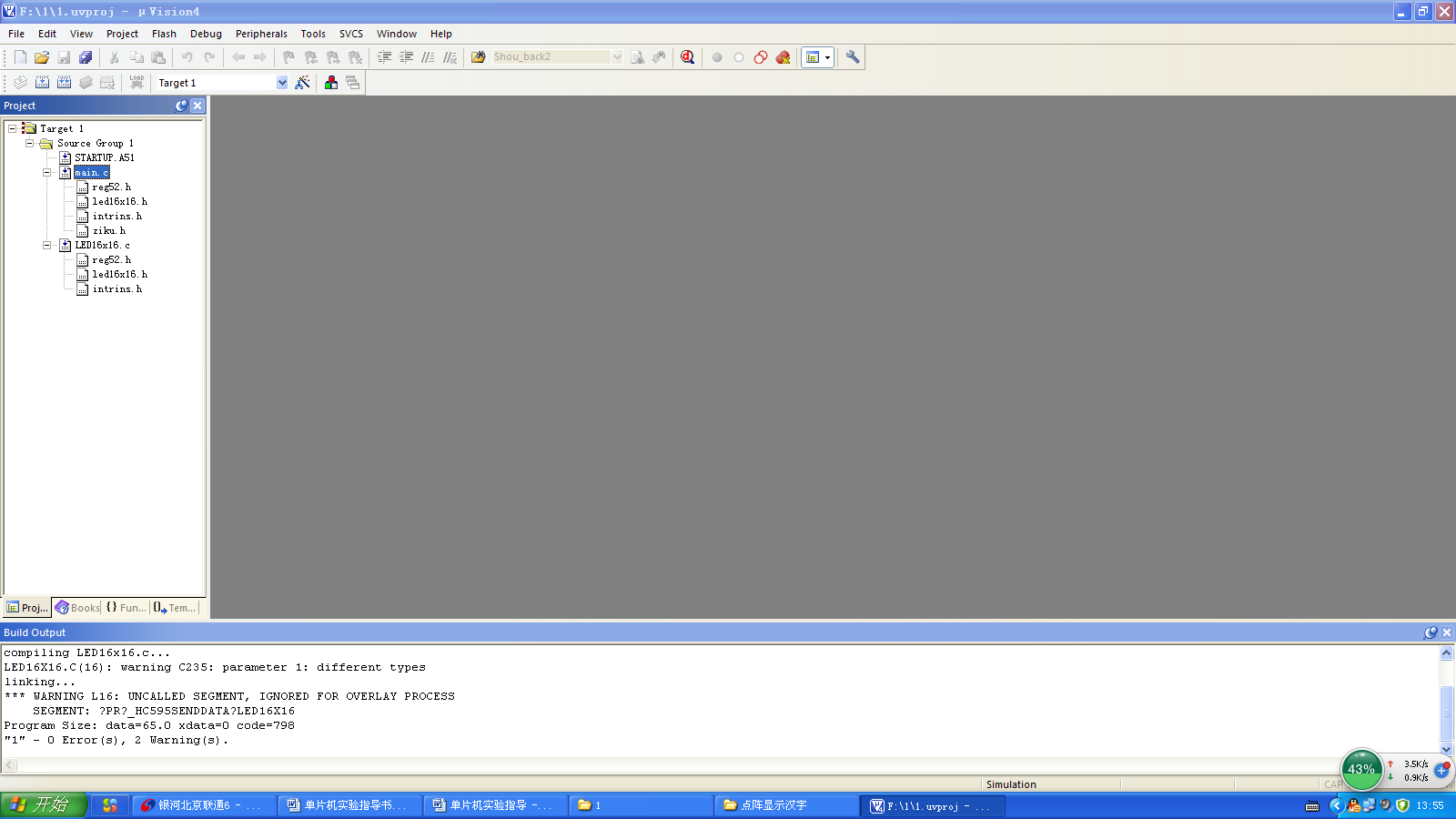
1、在可写入的F盘，新建一个工作文件夹，例如：文件夹名为“1”；

2、双击“”，进入“Keil uVision4”软件环境；

3、选择菜单栏中的“Project”项目中“New uVision Project…”选项，命名建立一个\*.pjt工程项目，弹出“Select a CPU Data Base File”窗口中，选择目标芯片系列：Generic CPU Data Base，点击“OK”；在弹出窗口“Data Base”项目中，选择“Atmel”，再选择“AT89C51”芯片，点击“OK”，弹出“Copy ‘STARTUP.A51’to Project Folder and Add File to Project”窗口中，选择“是”，项目建立完成。

4、将主窗口左侧工程管理窗口中“Target 1”的“+”点开，选择菜单栏中的“File”项目中“New”，根据类型输入源程序后另存为\*.h或 \*.c到\*.pjt所在的文件夹根目录下，注意后缀名.h及.c需要手动输入。项目中所需文件建立完成。

5、选择主窗口左侧工程管理窗口中“Target 1”的“+”点开，鼠标右键“Source Group”，选择菜单中“Add Files to Group ‘Soucre Group 1’ …”，弹出窗口中选择刚才保存的\*.c文件（注意\*.h不能添加），点击“Add”，然后点击“Close”，关闭该窗口。项目组建完成。

6、点击快捷工具栏中“”（Rebuild），对照报错窗口中的提示改错，当报错窗口中的错误（Errors）、警告（Warnings）均无误后，出现提示信息：0 Error（s）, 0 Warning（s），项目生成结果文件\*.hex（注意：点击“Project”中“Options for Target”，出现窗口点击“output”，勾选“Create HEX File”）。

7、下载程序结果，运用STC-ISP软件独立下载\*.hex结果文件到单片机教学系统。

8、运行程序结果，在教学系统对应的显示模块中出现相应的现象。

**五、写出实验源程序**

**实验（一）**

**//led16x16.h**

**#ifndef \_\_LED16X16\_H\_\_**

**#define \_\_LED16X16\_H\_\_**

**#include <intrins.h>**

**#include<reg52.h>**

**#define COW1 (0<<4)**

**#define COW2 (1<<4)**

**#define COW3 (2<<4)**

**#define COW4 (3<<4)**

**#define COW5 (4<<4)**

**#define COW6 (5<<4)**

**#define COW7 (6<<4)**

**#define COW8 (7<<4)**

**#define COW9 (8<<4)**

**#define COW10 (9<<4)**

**#define COW11 (10<<4)**

**#define COW12 (11<<4)**

**#define COW13 (12<<4)**

**#define COW14 (13<<4)**

**#define COW15 (14<<4)**

**#define COW16 (15<<4)**

**#define DATALONG 16**

**#define ADDRESS P2**

**#define NOP() // \_nop\_() /\* 定义空指令 \*/**

**sbit ADDA = P2^4;**

**sbit ADDB = P2^5;**

**sbit ADDC = P2^6;**

**sbit ADDD = P2^7;**

**sbit SHCP = P2^3;**

**sbit STCP = P2^2;**

**sbit DAIN = P2^1;**

**extern unsigned char mScanTable[16];**

**void HC595SendData(unsigned char SendVal);**

**void DISPLAY(unsigned int SendVal,unsigned char wei);**

**void sysShow();**

**void DrawDot(unsigned int \* Chat);**

**#endif**

**ziku.h**

**#ifndef \_\_ZIKU\_H\_**

**#define \_\_ZIKU\_H\_**

**unsigned char code ziku[]=**

**{**

**0x20,0x02,0x70,0x0A,0x1E,0x12,0x10,0x12,0x10,0x02,0xFF,0x7F,0x10,0x02,0x10,0x22,**

**0x50,0x22,0x30,0x12,0x18,0x0C,0x16,0x44,0x10,0x4A,0x10,0x51,0xD4,0x60,0x08,0x40,/\*"?",0\*/**

**0x00,0x10,0x80,0x3F,0x7E,0x08,0x44,0x08,0x88,0x04,0xFE,0x7F,0x42,0x40,0x41,0x20,**

**0xFE,0x1F,0x20,0x00,0xE0,0x0F,0x50,0x08,0x88,0x04,0x04,0x03,0xC2,0x0C,0x38,0x70,/\*"?",1\*/**

**0x08,0x08,0x10,0x04,0x20,0x02,0xFC,0x1F,0x84,0x10,0x84,0x10,0xFC,0x1F,0x84,0x10,**

**0x84,0x10,0xFC,0x1F,0x80,0x00,0x80,0x00,0xFF,0x7F,0x80,0x00,0x80,0x00,0x80,0x00,/\*"?",2\*/**

**0x00,0x02,0x08,0x02,0x08,0x02,0x08,0x02,0x08,0x02,0xF8,0x3F,0x08,0x00,0x08,0x00,**

**0x08,0x00,0xF8,0x07,0x08,0x04,0x08,0x04,0x08,0x04,0x04,0x04,0x04,0x04,0x02,0x04,/\*"?",3\*/**

**0x08,0x00,0x88,0x0F,0x88,0x08,0x88,0x08,0xBF,0x08,0x88,0x08,0x8C,0x08,0x9C,0x08,**

**0xAA,0x08,0xAA,0x08,0x89,0x08,0x88,0x48,0x88,0x48,0x48,0x48,0x48,0x70,0x28,0x00,/\*"?",4\*/**

**0x00,0x00,0x08,0x00,0x08,0x00,0x08,0x00,0x08,0x00,0x08,0x00,0x08,0x00,0x08,0x00,**

**0x08,0x00,0x08,0x00,0x00,0x00,0x00,0x00,0x08,0x00,0x08,0x00,0x00,0x00,0x00,0x00,/\*"!",5\*/**

**};**

**#endif**

**//LED16x16.c**

**#include "reg52.h"**

**#include "LED16x16.h"**

**unsigned char mScanTable[16]={**

**COW1,COW2,COW3,COW4,**

**COW5,COW6,COW7,COW8,**

**COW9,COW10,COW11,COW12,**

**COW13,COW14,COW15,COW16**

**};**

**unsigned int Buffer[16];**

**void HC595SendData(unsigned int SendVal)**

**{**

**unsigned char i;**

**for(i=0;i<DATALONG;i++)**

**{**

**if((SendVal<<i)&0x8000)**

**DAIN=0;**

**else**

**DAIN=1;**

**SHCP=0;**

**NOP();**

**NOP();**

**SHCP=1;**

**}**

**STCP=0; //set dataline low**

**NOP();**

**NOP();**

**}**

**void DISPLAY(unsigned int SendVal,unsigned char wei)**

**{**

**unsigned char i;**

**for(i=0;i<DATALONG;i++)**

**{**

**if((SendVal<<i)&0x8000)**

**DAIN=0;**

**else**

**DAIN=1;**

**SHCP=0;**

**NOP();**

**NOP();**

**SHCP=1;**

**}**

**STCP=0; //set dataline low**

**NOP();**

**NOP();**

**ADDRESS=(ADDRESS&0x0f)|mScanTable[wei];**

**STCP=1;**

**}**

**unsigned char mCow=0;**

**void DrawDot(unsigned int \* Chat)**

**{**

**unsigned char i=0;**

**for(i=0;i<16;i++)**

**{**

**Buffer[i]=\*Chat++;**

**}**

**}**

**void sysShow()**

**{**

**DISPLAY(Buffer[mCow],mCow);**

**mCow++;**

**mCow&=0x0f;**

**DISPLAY(Buffer[mCow],mCow);**

**mCow++;**

**mCow&=0x0f;**

**}**

**//main.c**

**#include "reg52.h"**

**#include "led16x16.h"**

**#include "ziku.h"**

**void delays(unsigned int time)**

**{**

**unsigned int i,j;**

**for(i=0;i<time;i++)**

**for(j-0;j<1000;j++);**

**}**

**void system\_Ini()**

**{**

**TMOD= 0x21;**

**TH0 = (65536-30000)>>8; //12.000**

**TL0 = (65536-30000)&0xff;**

**ET0 =1;**

**TR0 = 1;**

**SCON = 0x50; //REN=1允许串行接受状态，串口工作模式2**

**TMOD|= 0x20; //定时器工作方式2**

**PCON|= 0x80; //波特率提高一倍**

**TH1 = 0xF3;**

**TL1 = 0xF3;**

**TR1 = 1; //开启定时器1**

**ES = 1; //开串口中断**

**// IT0=0; //低电平触发**

**IT0=1; //下降沿触发**

**EX0=1;**

**EA = 1; // 开总中断**

**}**

**void main(void)**

**{**

**unsigned char i=0;**

**unsigned char \* CH=ziku;**

**system\_Ini();**

**while(1)**

**{**

**CH=ziku;**

**for(i=0;i<80;i++)**

**{**

**DrawDot((unsigned int \*)CH);**

**delays(500);**

**CH+=2;**

**if(!(i&0x0f))**

**delays(3900);**

**}**

**}**

**}**

**/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**[ t1 (1ms)中断] 中断**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/**

**void T1zd(void) interrupt 1 //3定时器0的中断号 1定时器0的中断号 0外部中断1 2外部中断2 4串口中断**

**{**

**TH0 = (65536-3000)>>8; //12.000**

**TL0 = (65536-3000)&0xff;**

**sysShow();**

**}**

**实验（二）**

**六、针对本实验写出实验心得**

**通过本次实验了解了单片机和LED的工作原理、I/O访问方式、单片机硬件制式，熟悉了Keil uVision4和STC-ISP软件的使用、程序编译和烧录方式。在烧录时应注意串口连线和对串口的保护，避免错误操作导致串口烧毁。**

**实验四 12864液晶屏实验**

**一、实验目的**

**1、了解12864液晶屏的接口定义及使用**

**2、熟悉单片机控制12864液晶屏的程序设计**

**3、**

**二、实验设备**

计算机，STC单片机下载线，单片机教学系统。

**三、实验原理**

1、LCD是英文Liquid Crystal Display 的简写，即为液晶显示。

2、带中文字库的12864液晶屏是一种具有4位/8位并行、2线或3线串行多种接口方式，内部含有国标一级、二级简体中文字库的点阵图形液晶显示模块；其显示分辨率为128×64, 内置8192个16\*16点汉字，和128个16\*8点ASCII字符集.利用该模块灵活的接口方式和简单、方便的操作指令，可构成全中文人机交互图形界面。可以显示8×4行16×16点阵的汉字，也可完成图形显示。低电压低功耗是其又一显著特点。

3、本实验是由单片机通过I/O接口对12864液晶屏进行显示控制，在可显示范围按要求显示。

**四、实验步骤**

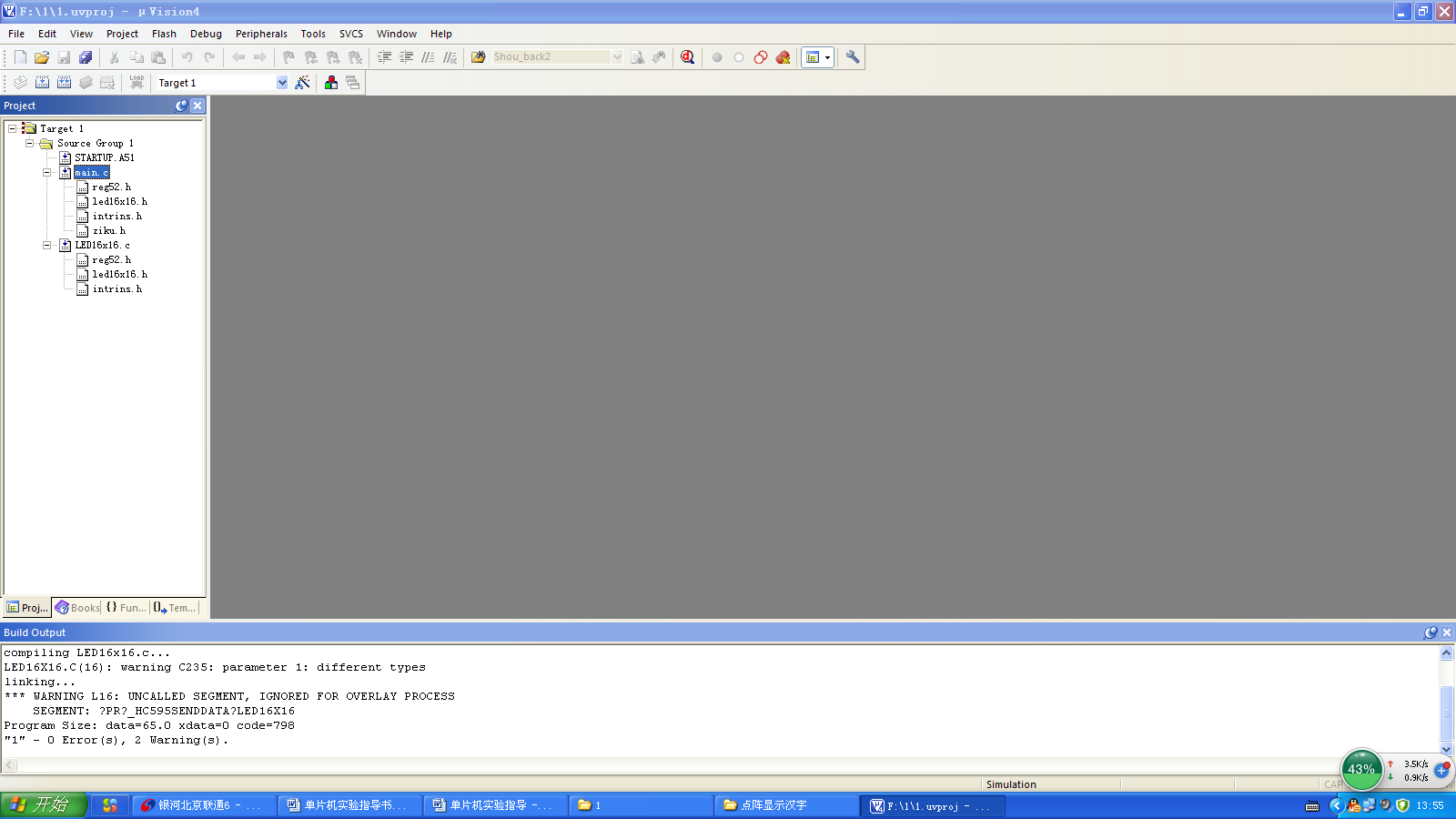
1、在可写入的F盘，新建一个工作文件夹，例如：文件夹名为“1”；

2、双击“”，进入“Keil uVision4”软件环境；

3、选择菜单栏中的“Project”项目中“New uVision Project…”选项，命名建立一个\*.pjt工程项目，弹出“Select a CPU Data Base File”窗口中，选择目标芯片系列：Generic CPU Data Base，点击“OK”；在弹出窗口“Data Base”项目中，选择“Atmel”，再选择“AT89C51”芯片，点击“OK”，弹出“Copy ‘STARTUP.A51’to Project Folder and Add File to Project”窗口中，选择“是”，项目建立完成。

4、将主窗口左侧工程管理窗口中“Target 1”的“+”点开，选择菜单栏中的“File”项目中“New”，根据类型输入源程序后另存为\*.h或 \*.c到\*.pjt所在的文件夹根目录下，注意后缀名.h及.c需要手动输入。项目中所需文件建立完成。

5、选择主窗口左侧工程管理窗口中“Target 1”的“+”点开，鼠标右键“Source Group”，选择菜单中“Add Files to Group ‘Soucre Group 1’ …”，弹出窗口中选择刚才保存的\*.c文件（注意\*.h不能添加），点击“Add”，然后点击“Close”，关闭该窗口。项目组建完成。

6、点击快捷工具栏中“”（Rebuild），对照报错窗口中的提示改错，当报错窗口中的错误（Errors）、警告（Warnings）均无误后，出现提示信息：0 Error（s）, 0 Warning（s），项目生成结果文件\*.hex（注意：点击“Project”中“Options for Target”，出现窗口点击“output”，勾选“Create HEX File”）。

7、下载程序结果，运用STC-ISP软件独立下载\*.hex结果文件到单片机教学系统。

8、运行程序结果，在教学系统对应的显示模块中出现相应的现象。

**五、写出实验源程序**

**实验（一）**

**//picture.h**

**#ifndef \_\_PICTURE\_H\_\_**

**#define \_\_PICTURE\_H\_\_**

**const unsigned char code Image\_pic[1];**

**#endif**

**//main.h**

**#ifndef \_\_MAIN\_h\_\_**

**#define \_\_MAIN\_h\_\_**

**typedef unsigned char uint8;**

**typedef signed char int8;**

**typedef unsigned int uint16;**

**typedef signed int int16;**

**typedef unsigned long uint32;**

**typedef signed long int32;**

**extern xdata unsigned char buffer[512];**

**#define ROW 480 //显示的行、列数**

**#define COL 800**

**#define DBH P1 //接口定义部分**

**#define DBL P0**

**sbit CS0 = P2^7;**

**sbit RST = P2^3;**

**sbit RS = P2^6;**

**sbit WR0 = P2^5;**

**sbit RD0 = P2^4;**

**#ifdef READ\_REG**

**#define STRING\_FUNCTION**

**#define READ\_FUNCTION**

**#endif**

**#ifdef DEBUG**

**#define STRING\_FUNCTION**

**#endif**

**#ifdef CONN\_USB**

**#define STRING\_FUNCTION**

**#define READ\_FUNCTION**

**#endif**

**#ifdef TP\_TEST**

**#define DRAW\_FUNCTION**

**#define STRING\_FUNCTION**

**#endif**

**#ifdef AUTO\_TEST**

**#define AUTO\_NEXT**

**#else**

**#define MANUA\_NEXT**

**#endif**

**#ifdef CHAR\_FONT\_W8\_H16**

**#define FONT\_W 8**

**#define FONT\_H 16**

**#endif**

**#ifdef CHAR\_FONT\_W16\_H21**

**#define FONT\_W 16**

**#define FONT\_H 21**

**#endif**

**#define PIC\_WIDTH 160 //预备向LCD显示区域填充的图片的大小**

**#define PIC\_HEIGHT 160**

**#define RED 0xF800 //定义颜色常量**

**#define GREEN 0x07E0**

**#define BLUE 0x001F**

**#define WHITE 0xFFFF**

**#define BLACK 0x0000**

**#define GRAY 0xEF5D //0x2410**

**#define GRAY75 0x39E7**

**#define GRAY50 0x7BEF**

**#define GRAY25 0xADB5**

**#define X\_min 0x0043 //TP测试范围常量定义**

**#define X\_max 0x07AE**

**#define Y\_min 0x00A1**

**#define Y\_max 0x0759**

**void Delay(unsigned int dly);**

**void StopDelay(unsigned int sdly);**

**void WaitKey(void);**

**void ConnToUSB(void);**

**void Suspend\_Device(void);**

**void Devices\_Init(void);**

**void MCU\_Init(void);**

**#endif**

**//lcd.h**

**#include<reg52.h>**

**#include <main.h>**

**#ifndef \_\_LCD\_h\_\_**

**#define \_\_LCD\_h\_\_**

**#ifdef DEBUG**

**#define END 0xffff**

**#endif**

**extern unsigned int code pic\_eval[];**

**extern unsigned char code ascii[];**

**extern unsigned int code debug\_code[50][15];**

**void WriteComm(unsigned int i);**

**void WriteData(unsigned int i);**

**void WriteDispData(unsigned char DataH,unsigned char DataL);**

**void LCD\_Init(void);**

**void BlockWrites(unsigned int Xstart,unsigned int Xend,unsigned int Ystart,unsigned int Yend);**

**void DispBand(void);**

**void DispColor(unsigned int StartX,unsigned int StartY,unsigned int Col,unsigned int Row,unsigned int color);**

**void DispFrame(void);**

**void DispPic(unsigned int code \*picture);**

**void DisSin(unsigned char code \*table,unsigned char lenght,unsigned char freq,unsigned int color);**

**void InterBresenhamline (unsigned int x1,unsigned int y1,unsigned int x2,unsigned int y2,unsigned int doc);**

**void DispScaleHor1(void);**

**void DispScaleVer(void);**

**void DispScaleVer\_Red(void);**

**void DispScaleVer\_Green(void);**

**void DispScaleVer\_Blue(void);**

**void DispScaleVer\_Gray(void);**

**void DispGrayHor16(void);**

**void DispGrayHor32(void);**

**void DispScaleHor2(void);**

**void DispSnow(void);**

**void DispBlock(void);**

**void WriteOneDot(unsigned int color);**

**unsigned char ToOrd(unsigned char ch);**

**void DispOneChar(unsigned char ord,unsigned int Xstart,unsigned int Ystart,unsigned int TextColor,unsigned int BackColor);**

**void DispStr(unsigned char \*str,unsigned int Xstart,unsigned int Ystart,unsigned int TextColor,unsigned int BackColor);**

**void DispInt(unsigned int i,unsigned int Xstart,unsigned int Ystart,unsigned int TextColor,unsigned int BackColor);**

**void MTP\_ADJ(void);**

**void grayscale(void);**

**unsigned int ReadData(void);**

**void DispRegValue(unsigned int RegIndex,unsigned char ParNum);**

**void Debug(void);**

**void PutPixel(unsigned int x,unsigned int y,unsigned int color);**

**void DrawLine(unsigned int Xstart,unsigned int Xend,unsigned int Ystart,unsigned int Yend,unsigned int color);**

**void DrawGird(unsigned int color);**

**#endif**

**//lcd.c**

**#include<reg52.h>**

**#include <intrins.h>**

**#include <absacc.h>**

**#include <main.h>**

**#include <lcd\lcd.h>**

**#define DRAW\_FUNCTION**

**#define EVAL\_PIC**

**unsigned int ID1,ID2,ID3,ram,LCDrd\_data,VCOM,temp=0x1a00;**

**void WriteComm(unsigned int i)**

**{**

**CS0=0;**

**RD0=1;**

**RS=0;**

**DBH=i>>8;**

**DBL=i;**

**WR0=0;**

**WR0=1;**

**CS0=1;**

**}**

**void WriteData(unsigned int i)**

**{**

**CS0=0;**

**RD0=1;**

**RS=1;**

**DBH=i>>8;**

**DBL=i;**

**WR0=0;**

**WR0=1;**

**CS0=1;**

**}**

**void WriteOneDot(unsigned int color)**

**{**

**CS0=0;**

**RD0=1;**

**RS=1;**

**DBH=color>>8;**

**DBL=color;**

**WR0=0;**

**WR0=1;**

**CS0=1;**

**}**

**void LCD\_Init(void)**

**{**

**Delay(1000);**

**RST=0;**

**Delay(10000);**

**RST=1;**

**Delay(10000);**

**WriteComm(0x11);**

**Delay(200);**

**WriteComm(0xB0);**

**WriteData(0x04);**

**WriteComm(0xB3);//Frame Memory Access and Interface Setting**

**WriteData(0x02);**

**WriteData(0x00);**

**WriteComm(0xC1);//Panel Driving Setting**

**WriteData(0x23);**

**WriteData(0x31);//NL**

**WriteData(0x99);**

**WriteData(0x21);**

**WriteData(0x20);**

**WriteData(0x00);**

**WriteData(0x10);//DIVI**

**WriteData(0x28);//RTN**

**WriteData(0x0C);//BP**

**WriteData(0x0A);//FP**

**WriteData(0x00);**

**WriteData(0x00);**

**WriteData(0x00);**

**WriteData(0x21);**

**WriteData(0x01);**

**WriteComm(0xC2);//Display V-Timing Setting**

**WriteData(0x00);**

**WriteData(0x06);**

**WriteData(0x06);**

**WriteData(0x01);**

**WriteData(0x03);**

**WriteData(0x00);**

**WriteComm(0xC8);//GAMMA**

**WriteData(0x01);**

**WriteData(0x0A);**

**WriteData(0x12);**

**WriteData(0x1C);**

**WriteData(0x2B);**

**WriteData(0x45);**

**WriteData(0x3F);**

**WriteData(0x29);**

**WriteData(0x17);**

**WriteData(0x13);**

**WriteData(0x0F);**

**WriteData(0x04);**

**WriteData(0x01);**

**WriteData(0x0A);**

**WriteData(0x12);**

**WriteData(0x1C);**

**WriteData(0x2B);**

**WriteData(0x45);**

**WriteData(0x3F);**

**WriteData(0x29);**

**WriteData(0x17);**

**WriteData(0x13);**

**WriteData(0x0F);**

**WriteData(0x04);**

**WriteComm(0xC9);//GAMMA**

**WriteData(0x01);**

**WriteData(0x0A);**

**WriteData(0x12);**

**WriteData(0x1C);**

**WriteData(0x2B);**

**WriteData(0x45);**

**WriteData(0x3F);**

**WriteData(0x29);**

**WriteData(0x17);**

**WriteData(0x13);**

**WriteData(0x0F);**

**WriteData(0x04);**

**WriteData(0x01);**

**WriteData(0x0A);**

**WriteData(0x12);**

**WriteData(0x1C);**

**WriteData(0x2B);**

**WriteData(0x45);**

**WriteData(0x3F);**

**WriteData(0x29);**

**WriteData(0x17);**

**WriteData(0x13);**

**WriteData(0x0F);**

**WriteData(0x04);**

**WriteComm(0xCA);//GAMMA**

**WriteData(0x01);**

**WriteData(0x0A);**

**WriteData(0x12);**

**WriteData(0x1C);**

**WriteData(0x2B);**

**WriteData(0x45);**

**WriteData(0x3F);**

**WriteData(0x29);**

**WriteData(0x17);**

**WriteData(0x13);**

**WriteData(0x0F);**

**WriteData(0x04);**

**WriteData(0x01);**

**WriteData(0x0A);**

**WriteData(0x12);**

**WriteData(0x1C);**

**WriteData(0x2B);**

**WriteData(0x45);**

**WriteData(0x3F);**

**WriteData(0x29);**

**WriteData(0x17);**

**WriteData(0x13);**

**WriteData(0x0F);**

**WriteData(0x04);**

**WriteComm(0xD0);//Power Setting (Charge Pump Setting)**

**WriteData(0x99);//DC**

**WriteData(0x03);**

**WriteData(0xCE);**

**WriteData(0xA6);**

**WriteData(0x00);//CP or SR**

**WriteData(0x43);//VC3, VC2**

**WriteData(0x20);**

**WriteData(0x10);**

**WriteData(0x01);**

**WriteData(0x00);**

**WriteData(0x01);**

**WriteData(0x01);**

**WriteData(0x00);**

**WriteData(0x03);**

**WriteData(0x01);**

**WriteData(0x00);**

**WriteComm(0xD3);//Power Setting for Internal Mode**

**WriteData(0x33);//AP**

**WriteComm(0xD5);//VPLVL/VNLVL Setting**

**WriteData(0x2A);**

**WriteData(0x2A);**

**WriteComm(0xD6);**

**WriteData(0xA8);**

**WriteComm(0xD6);**

**WriteData(0x01);**

**WriteComm(0xDE);//VCOMDC Setting**

**WriteData(0x01);**

**WriteData(0x4F);**

**WriteComm(0xE6);//VCOMDC Setting**

**WriteData(0x4F);**

**WriteComm(0xFA);//VDC\_SEL Setting**

**WriteData(0x03);**

**Delay(100);**

**WriteComm(0x2A);**

**WriteData(0x00);**

**WriteData(0x00);**

**WriteData(0x01);**

**WriteData(0xDF);**

**WriteComm(0x2B);**

**WriteData(0x00);**

**WriteData(0x00);**

**WriteData(0x03);**

**WriteData(0x1F);**

**WriteComm(0x36);**

**WriteData(0x60);**

**WriteComm(0x3A);**

**WriteData(0x55);**

**WriteComm(0x29);**

**Delay(20);**

**WriteComm(0x2C);**

**Delay(10);**

**DispColor(0,0,COL,ROW,WHITE);**

**Delay(20);**

**}**

**void BlockWrites(unsigned int Xstart,unsigned int Xend,unsigned int Ystart,unsigned int Yend) //reentrant**

**{**

**WriteComm(0x2a); //Lcd\_Delay(10);**

**WriteData(Xstart>>8);//Lcd\_Delay(10);**

**WriteData(Xstart&0xff);//Lcd\_Delay(10);**

**WriteData(Xend>>8);//Lcd\_Delay(10);**

**WriteData(Xend&0xff);//Lcd\_Delay(10);**

**WriteComm(0x2b); //Lcd\_Delay(10);**

**WriteData(Ystart>>8);//Lcd\_Delay(10);**

**WriteData(Ystart&0xff);//Lcd\_Delay(10);**

**WriteData(Yend>>8);//Lcd\_Delay(10);**

**WriteData(Yend&0xff);//Lcd\_Delay(10);**

**WriteComm(0x2c);//Lcd\_Delay(10);**

**}**

**void DispColor(unsigned int StartX,unsigned int StartY,unsigned int Col,unsigned int Row,unsigned int color)**

**{**

**unsigned int i,j;**

**BlockWrites(StartX,Col-1,StartY,Row-1);**

**CS0=0;**

**RS=1;**

**RD0=1;**

**DBH=color>>8;**

**DBL=color;**

**for(i=0;i<Row;i++)**

**{**

**for(j=0;j<Col;j++)**

**{**

**WR0=0;**

**WR0=1;//WR0=0;**

**}**

**}**

**}**

**void DisSin(unsigned char code \*table,unsigned char lenght,unsigned char freq,unsigned int color)**

**{**

**unsigned int X=20,Ystart,Ystop;**

**unsigned char \*ch=table;**

**unsigned char Len=lenght;**

**Ystart=\*ch++;**

**Ystop=\*ch++;**

**Len--;**

**Len--;**

**while(X<620)**

**{**

**InterBresenhamline(X,Ystart+70,((X+freq)>620)? 620:(X+freq),Ystop+70,color);**

**X+=freq;**

**if(Len==1)**

**{**

**ch=table;**

**Len=lenght;**

**}**

**Ystart=Ystop;**

**Ystop=\*ch++;**

**Len--;**

**}**

**}**

**void DispBand(void)**

**{**

**unsigned int i,j,k;**

**};**

**unsigned int color[8]={0xf800,0xf800,0x07e0,0x07e0,0x001f,0x001f,0xffff,0xffff};//0x94B2**

**BlockWrites(0,COL-1,0,ROW-1);**

**CS0=0;**

**RD0=1;**

**RS=1;**

**for(i=0;i<8;i++)**

**{**

**for(j=0;j<ROW/8;j++)**

**{**

**for(k=0;k<COL;k++)**

**{**

**DBL=color[i];**

**DBH=color[i]>>8;**

**WR0=0;**

**WR0=1;**

**}**

**}**

**}**

**for(j=0;j<ROW%8;j++)**

**{**

**for(k=0;k<COL;k++)**

**{**

**DBL=color[7];**

**DBH=color[7]>>8; //Lcd\_Delay(10);**

**WR0=0; //Lcd\_Delay(10);**

**WR0=1; //Lcd\_Delay(10);**

**}**

**}**

**CS0=1;**

**}**

**#ifdef EVAL\_PIC**

**void DispPic(unsigned int code \*picture)**

**{**

**unsigned int \*p;**

**unsigned int i,j; //i-row,j-col**

**unsigned int n,k; //n-row repeat count,k-col repeat count**

**BlockWrites(0,COL-1,0,ROW-1);**

**CS0=0;**

**RS =1;**

**RD0=1;**

**for(n=0;n<ROW/PIC\_HEIGHT;n++) //n-row repeat count**

**{**

**for(i=0;i<PIC\_HEIGHT;i++)**

**{**

**p=picture;**

**for(k=0;k<COL/PIC\_WIDTH;k++) //k-col repeat count**

**{**

**for(j=0;j<PIC\_WIDTH;j++)**

**{**

**DBH=(\*(p+i\*PIC\_HEIGHT+j))>>8;**

**//WR0=0;**

**//WR0=1;**

**DBL=\*(p+i\*PIC\_HEIGHT+j);**

**WR0=0;**

**WR0=1;**

**}**

**}**

**p=picture;**

**for(j=0;j<COL%PIC\_WIDTH;j++)**

**{**

**DBH=(\*(p+i\*PIC\_HEIGHT+j))>>8;**

**//WR0=0;**

**//WR0=1;**

**DBL=\*(p+i\*PIC\_HEIGHT+j);**

**WR0=0;**

**WR0=1;**

**}**

**}**

**}**

**for(i=0;i<ROW%PIC\_HEIGHT;i++)**

**{**

**p=picture;**

**for(k=0;k<COL/PIC\_WIDTH;k++) //k-col repeat count**

**{**

**for(j=0;j<PIC\_WIDTH;j++)**

**{**

**DBH=(\*(p+i\*PIC\_HEIGHT+j))>>8;**

**DBL=\*(p+i\*PIC\_HEIGHT+j);**

**WR0=0;**

**WR0=1;**

**}**

**}**

**p=picture;**

**for(j=0;j<COL%PIC\_WIDTH;j++)**

**{**

**DBH=(\*(p+i\*PIC\_HEIGHT+j))>>8;**

**DBL=\*(p+i\*PIC\_HEIGHT+j);**

**WR0=0;**

**WR0=1;**

**}**

**}**

**CS0=1;**

**}**

**#endif**

**#ifdef EXTEND\_TEST**

**void DispScaleHor1(void)**

**{**

**unsigned int i,j,k;**

**BlockWrites(0,COL-1,0,ROW-1);**

**CS0=0;**

**RD0=1;**

**RS=1;**

**for(k=0;k<ROW/4;k++)**

**{**

**for(i=0;i<COL%32;i++){DBH=0x00,DBL=0x00,WR0=0;WR0=1;}**

**for(i=0;i<32;i++)**

**{**

**for(j=0;j<COL/32;j++)**

**{**

**DBH=i<<3;**

**DBL=0;**

**WR0=0;**

**WR0=1;**

**}**

**}**

**}**

**for(k=0;k<ROW/4;k++)**

**{**

**for(i=0;i<COL%32;i++){DBH=0x00,DBL=0x00,WR0=0;WR0=1;}**

**for(i=0;i<32;i++)**

**{**

**for(j=0;j<COL/32;j++)**

**{**

**DBH=(i\*2)>>3;**

**DBL=(i\*2)<<5;**

**WR0=0;**

**WR0=1;**

**}**

**}**

**}**

**for(k=0;k<ROW/4;k++)**

**{**

**for(i=0;i<COL%32;i++){DBH=0x00,DBL=0x00,WR0=0;WR0=1;}**

**for(i=0;i<32;i++)**

**{**

**for(j=0;j<COL/32;j++)**

**{**

**DBH=0;**

**DBL=i;**

**WR0=0;**

**WR0=1;**

**}**

**}**

**}**

**for(k=0;k<(ROW/4+ROW%4);k++)**

**{**

**for(i=0;i<COL%32;i++){DBH=0x00,DBL=0x00,WR0=0;WR0=1;}**

**for(i=0;i<32;i++)**

**{**

**for(j=0;j<COL/32;j++)**

**{**

**DBH=(i<<3)|((i\*2)>>3);**

**DBL=((i\*2)<<5)|i;**

**WR0=0;**

**WR0=1;**

**}**

**}**

**}**

**CS0=1;**

**}**

**void DispScaleVer(void)**

**{**

**unsigned int i,j,k;**

**BlockWrites(0,COL-1,0,ROW-1);**

**CS0=0;**

**RD0=1;**

**RS=1;**

**for(k=0;k<ROW%32;k++)**

**{**

**for(j=0;j<COL;j++)**

**{**

**DBH=0;**

**DBL=0;**

**WR0=0;**

**WR0=1;**

**}**

**}**

**for(k=0;k<32;k++)**

**{**

**for(i=0;i<ROW/32;i++)**

**{**

**for(j=0;j<COL/4;j++)**

**{**

**DBH=k<<3;**

**DBL=0;**

**WR0=0;**

**WR0=1;**

**}**

**for(j=0;j<COL/4;j++)**

**{**

**DBH=(k\*2)>>3;**

**DBL=(k\*2)<<5;**

**WR0=0;**

**WR0=1;**

**}**

**for(j=0;j<COL/4;j++)**

**{**

**DBH=0;**

**DBL=k;**

**WR0=0;**

**WR0=1;**

**}**

**for(j=0;j<COL/4;j++)**

**{**

**DBH=(k<<3)|((k\*2)>>3);**

**DBL=((k\*2)<<5)|k;**

**WR0=0;**

**WR0=1;**

**}**

**}**

**}**

**CS0=1;**

**}**

**void DispScaleVer\_Red(void)**

**{**

**unsigned int i,j,k;**

**BlockWrites(0,COL-1,0,ROW-1);**

**CS0=0;**

**RD0=1;**

**RS=1;**

**for(k=0;k<ROW%32;k++)**

**{**

**for(j=0;j<COL;j++)**

**{**

**DBH=0;**

**DBL=0;**

**WR0=0;**

**WR0=1;**

**}**

**}**

**for(k=0;k<32;k++)**

**{**

**for(i=0;i<ROW/32;i++)**

**{**

**for(j=0;j<COL;j++)**

**{**

**DBH=k<<3;**

**DBL=0;**

**WR0=0;**

**WR0=1;**

**}**

**}**

**}**

**CS0=1;**

**}**

**void DispScaleVer\_Green(void)**

**{**

**unsigned int i,j,k;**

**BlockWrites(0,COL-1,0,ROW-1);**

**CS0=0;**

**RD0=1;**

**RS=1;**

**for(k=0;k<ROW%64;k++)**

**{**

**for(j=0;j<COL;j++)**

**{**

**DBH=0;**

**DBL=0;**

**WR0=0;**

**WR0=1;**

**}**

**}**

**for(k=0;k<64;k++)**

**{**

**for(i=0;i<ROW/64;i++)**

**{**

**for(j=0;j<COL;j++)**

**{**

**DBH=k>>3;**

**DBL=k<<5;**

**WR0=0;**

**WR0=1;**

**}**

**}**

**}**

**CS0=1;**

**}**

**void DispScaleVer\_Blue(void)**

**{**

**unsigned int i,j,k;**

**BlockWrites(0,COL-1,0,ROW-1);**

**CS0=0;**

**RD0=1;**

**RS=1;**

**for(k=0;k<ROW%32;k++)**

**{**

**for(j=0;j<COL;j++)**

**{**

**DBH=0;**

**DBL=0;**

**WR0=0;**

**WR0=1;**

**}**

**}**

**for(k=0;k<32;k++)**

**{**

**for(i=0;i<ROW/32;i++)**

**{**

**for(j=0;j<COL;j++)**

**{**

**DBH=0;**

**DBL=k;**

**WR0=0;**

**WR0=1;**

**}**

**}**

**}**

**CS0=1;**

**}**

**void DispScaleVer\_Gray(void)**

**{**

**unsigned int i,j,k;**

**BlockWrites(0,COL-1,0,ROW-1);**

**CS0=0;**

**RD0=1;**

**RS=1;**

**for(k=0;k<ROW%32;k++)**

**{**

**for(j=0;j<COL;j++)**

**{**

**DBH=0;**

**DBL=0;**

**WR0=0;**

**WR0=1;**

**}**

**}**

**for(k=0;k<32;k++)**

**{**

**for(i=0;i<ROW/32;i++)**

**{**

**for(j=0;j<COL;j++)**

**{**

**DBH=(k<<3)|((k\*2)>>3);**

**DBL=((k\*2)<<5)|k;**

**WR0=0;**

**WR0=1;**

**}**

**}**

**}**

**CS0=1;**

**}**

**void DispGrayHor16(void)**

**{**

**unsigned int i,j,k;**

**BlockWrites(0,COL-1,0,ROW-1);**

**CS0=0;**

**RD0=1;**

**RS=1;**

**for(i=0;i<ROW;i++)**

**{**

**for(j=0;j<COL%16;j++)**

**{**

**DBH=0;**

**DBL=0;**

**WR0=0;**

**WR0=1;**

**}**

**for(j=0;j<16;j++)**

**{**

**for(k=0;k<COL/16;k++)**

**{**

**DBH=((j\*2)<<3)|((j\*4)>>3);**

**DBL=((j\*4)<<5)|(j\*2);**

**WR0=0;**

**WR0=1;**

**}**

**}**

**}**

**CS0=1;**

**}**

**void DispGrayHor32(void)**

**{**

**unsigned int i,j,k;**

**BlockWrites(0,COL-1,0,ROW-1);**

**CS0=0;**

**RD0=1;**

**RS=1;**

**for(i=0;i<ROW;i++)**

**{**

**for(j=0;j<COL%32;j++)**

**{**

**DBH=0;**

**DBL=0;**

**WR0=0;**

**WR0=1;**

**}**

**for(j=0;j<32;j++)**

**{**

**for(k=0;k<COL/32;k++)**

**{**

**DBH=(j<<3)|((j\*2)>>3);**

**DBL=((j\*2)<<5)|j;**

**WR0=0;**

**WR0=1;**

**}**

**}**

**}**

**CS0=1;**

**}**

**void DispScaleHor2(void)**

**{**

**unsigned int i,j,k;**

**BlockWrites(0,COL-1,0,ROW-1);**

**CS0=0;**

**RD0=1;**

**RS=1;**

**for(k=0;k<ROW/6;k++)**

**{**

**for(i=0;i<COL%32;i++){DBH=0x00,DBL=0x00,WR0=0;WR0=1;}**

**for(i=0;i<32;i++)**

**{**

**for(j=0;j<COL/32;j++)**

**{**

**DBH=i<<3;**

**DBL=0;**

**WR0=0;**

**WR0=1;**

**}**

**}**

**}**

**for(k=0;k<ROW/6;k++)**

**{**

**for(i=32;i>0;i--)**

**{**

**for(j=0;j<COL/32;j++)**

**{**

**DBH=(i-1)<<3;**

**DBL=0;**

**WR0=0;**

**WR0=1;**

**}**

**}**

**for(i=0;i<COL%32;i++){DBH=0x00,DBL=0x00,WR0=0;WR0=1;}**

**}**

**for(k=0;k<ROW/6;k++)**

**{**

**for(i=0;i<COL%64;i++){DBH=0x00,DBL=0x00,WR0=0;WR0=1;}**

**for(i=0;i<64;i++)**

**{**

**for(j=0;j<COL/64;j++)**

**{**

**DBH=i>>3;**

**DBL=i<<5;**

**WR0=0;**

**WR0=1;**

**}**

**}**

**}**

**for(k=0;k<ROW/6;k++)**

**{**

**for(i=64;i>0;i--)**

**{**

**for(j=0;j<COL/64;j++)**

**{**

**DBH=(i-1)>>3;**

**DBL=(i-1)<<5;**

**WR0=0;**

**WR0=1;**

**}**

**}**

**for(i=0;i<COL%64;i++){DBH=0x00,DBL=0x00,WR0=0;WR0=1;}**

**}**

**for(k=0;k<ROW/6;k++)**

**{**

**for(i=0;i<COL%32;i++){DBH=0x00,DBL=0x00,WR0=0;WR0=1;}**

**for(i=0;i<32;i++)**

**{**

**for(j=0;j<COL/32;j++)**

**{**

**DBH=0;**

**DBL=i;**

**WR0=0;**

**WR0=1;**

**}**

**}**

**}**

**for(k=0;k<(ROW/6+ROW%6);k++)**

**{**

**for(i=32;i>0;i--)**

**{**

**for(j=0;j<COL/32;j++)**

**{**

**DBH=0;**

**DBL=(i-1);**

**WR0=0;**

**WR0=1;**

**}**

**}**

**for(i=0;i<COL%32;i++){DBH=0x00,DBL=0x00,WR0=0;WR0=1;}**

**}**

**CS0=1;**

**}**

**void DispSnow(void)**

**{**

**unsigned int i,j;**

**BlockWrites(0,COL-1,0,ROW-1);**

**CS0=0;**

**RS=1;**

**RD0=1;**

**DBL = 0;**

**DBH = 0;**

**for(i=0;i<ROW;i++)**

**{**

**for(j=0;j<COL/2;j++)**

**{**

**WR0 = 0; WR0 = 1;**

**DBL = ~DBL;**

**DBH = ~DBH;**

**WR0 = 0; WR0 = 1;**

**DBL = ~DBL;**

**DBH = ~DBH;**

**}**

**DBL = ~DBL;**

**DBH = ~DBH;**

**}**

**CS0 = 1;**

**}**

**void DispBlock(void)**

**{**

**unsigned int i,j,k;**

**k = ROW /4;**

**BlockWrites(0,COL-1,0,ROW-1);**

**CS0=0;**

**RS=1;**

**RD0=1;**

**DBH=0x7b;**

**DBL=0xef;**

**for(i=0;i<k;i++)**

**{**

**for(j=0;j<COL;j++)**

**{**

**WR0=0;**

**WR0=1;**

**}**

**}**

**for(i=0;i<k\*2;i++)**

**{**

**DBH=0x7b;**

**DBL=0xef;**

**for(j=0;j<COL/4;j++)**

**{**

**WR0=0;**

**WR0=1;**

**}**

**DBH=0x00;**

**DBL=0x00;**

**for(j=0;j<COL/2;j++)**

**{**

**WR0=0;**

**WR0=1;**

**}**

**DBH=0x7b;**

**DBL=0xef;**

**for(j=0;j<COL/4;j++)**

**{**

**WR0=0;**

**WR0=1;**

**}**

**}**

**DBH=0x7b;**

**DBL=0xef;**

**for(i=0;i<k;i++)**

**{**

**for(j=0;j<COL;j++)**

**{**

**WR0=0;**

**WR0=1;**

**}**

**}**

**CS0=1;**

**}**

**#endif**

**#ifdef STRING\_FUNCTION**

**unsigned char ToOrd(unsigned char ch)**

**{**

**if(ch<32)**

**{**

**ch=95;**

**}**

**else if((ch>=32)&&(ch<=47)) //(32~47)空格~/**

**{**

**ch=(ch-32)+10+62;**

**}**

**else if((ch>=48)&&(ch<=57))//(48~57)0~9**

**{**

**ch=ch-48;**

**}**

**else if((ch>=58)&&(ch<=64))//(58~64):~@**

**{**

**ch=(ch-58)+10+62+16;**

**}**

**else if((ch>=65)&&(ch<=126))//(65~126)A~~**

**{**

**ch=(ch-65)+10;**

**}**

**else if(ch>126)**

**{**

**ch=95;**

**}**

**return ch;**

**}**

**void DispOneChar(unsigned char ord,unsigned int Xstart,unsigned int Ystart,unsigned int TextColor,unsigned int BackColor) // ord:0~95**

**{**

**unsigned char i,j;**

**unsigned char \*p;**

**unsigned char dat;**

**unsigned int index;**

**BlockWrites(Xstart,Xstart+(FONT\_W-1),Ystart,Ystart+(FONT\_H-1));**

**index = ord;**

**if(index>95) //95:ASCII CHAR NUM**

**index=95;**

**index = index\*((FONT\_W/8)\*FONT\_H);**

**p = ascii;**

**p = p+index;**

**for(i=0;i<(FONT\_W/8\*FONT\_H);i++)**

**{**

**dat=\*p++;**

**for(j=0;j<8;j++)**

**{**

**if((dat<<j)&0x80)**

**{**

**WriteOneDot(TextColor);**

**}**

**else**

**{**

**WriteOneDot(BackColor);**

**}**

**}**

**}**

**}**

**void DispStr(unsigned char \*str,unsigned int Xstart,unsigned int Ystart,unsigned int TextColor,unsigned int BackColor)**

**{**

**while(!(\*str=='\0'))**

**{**

**DispOneChar(ToOrd(\*str++),Xstart,Ystart,TextColor,BackColor);**

**if(Xstart>((COL-1)-FONT\_W))**

**{**

**Xstart=0;**

**Ystart=Ystart+FONT\_H;**

**}**

**else**

**{**

**Xstart=Xstart+FONT\_W;**

**}**

**if(Ystart>((ROW-1)-FONT\_H))**

**{**

**Ystart=0;**

**}**

**}**

**BlockWrites(0,COL-1,0,ROW-1);**

**}**

**void DispInt(unsigned int i,unsigned int Xstart,unsigned int Ystart,unsigned int TextColor,unsigned int BackColor)**

**{**

**if(Xstart>((COL-1)-FONT\_W\*4))**

**{**

**Xstart=(COL-1)-FONT\_W\*4;**

**}**

**if(Ystart>((ROW-1)-FONT\_H))**

**{**

**Ystart=(Ystart-1)-FONT\_H;**

**}**

**DispOneChar((i>>12)%16,Xstart,Ystart,TextColor,BackColor); //ID value**

**DispOneChar((i>>8)%16,Xstart+FONT\_W,Ystart,TextColor,BackColor);**

**DispOneChar((i>>4)%16,Xstart+FONT\_W\*2,Ystart,TextColor,BackColor);**

**DispOneChar(i%16,Xstart+FONT\_W\*3,Ystart,TextColor,BackColor);**

**BlockWrites(0,COL-1,0,ROW-1);**

**}**

**#endif**

**#ifdef READ\_FUNCTION**

**unsigned int ReadData(void)**

**{**

**unsigned int temp;**

**DBL = 0x00;**

**DBH = 0xff;**

**CS0 = 0;**

**RS = 1;**

**WR0 = 1;**

**RD0 = 0;**

**\_nop\_(); \_nop\_();\_nop\_(); \_nop\_();**

**RD0 = 1;**

**\_nop\_(); \_nop\_();\_nop\_(); \_nop\_();**

**temp=DBH;**

**temp<<=8;**

**temp|=DBL;**

**RD0 = 1;**

**\_nop\_(); \_nop\_();\_nop\_(); \_nop\_();**

**CS0 = 1;**

**return temp;**

**}**

**#endif**

**#ifdef READ\_REG**

**void DispRegValue(unsigned int RegIndex,unsigned char ParNum)**

**{**

**unsigned char i;**

**xdata unsigned int reg\_data[20];**

**WriteComm(RegIndex); //ILI9481,BFh,P1~P6**

**if(ParNum>20)**

**ParNum=20;**

**for(i=0;i<ParNum;i++)**

**{**

**reg\_data[i]=ReadData();**

**}**

**DispStr("READ ID ",0,0,BLACK,WHITE);**

**for(i=0;i<ParNum;i++)**

**{**

**DispStr("0X",0,(FONT\_H+1)\*(i+1),BLUE,WHITE);**

**DispInt(reg\_data[i],FONT\_W\*2,(FONT\_H+1)\*(i+1),BLUE,WHITE);**

**}**

**}**

**#endif**

**#ifdef SLEEP\_TEST**

**void EnterSLP(void)**

**{**

**WriteComm(0x0007);WriteData(0x0131); // Set D1=0, D0=1**

**Delay(10);**

**WriteComm(0x0007);WriteData(0x0130); // Set D1=0, D0=0**

**Delay(10);**

**WriteComm(0x0007);WriteData(0x0000); // display OFF**

**WriteComm(0x0010);WriteData(0x0080);**

**WriteComm(0x0011);WriteData(0x0000);**

**WriteComm(0x0012);WriteData(0x0000); // VREG1OUT voltage**

**WriteComm(0x0013);WriteData(0x0000);**

**Delay(200); // Dis-charge capacitor power voltage**

**WriteComm(0x0010);WriteData(0x0082);**

**}**

**void ExitSLP(void)**

**{**

**WriteComm(0x0010);WriteData(0x0080);**

**WriteComm(0x0011);WriteData(0x0000);**

**WriteComm(0x0012);WriteData(0x0000); // VREG1OUT voltage**

**WriteComm(0x0013);WriteData(0x0000);**

**Delay(200); // Dis-charge capacitor power voltage**

**WriteComm(0x0010);WriteData(0x1490);**

**WriteComm(0x0011);WriteData(0x0227);**

**Delay(50); // Delay 50ms**

**WriteComm(0x0012);WriteData(0x009B);**

**Delay(50); // Delay 50ms**

**WriteComm(0x0013);WriteData(0x1800);**

**WriteComm(0x0029);WriteData(0x0027);**

**Delay(50); // Delay 50ms**

**WriteComm(0x0007);WriteData(0x0133);**

**}**

**#endif**

**#ifdef DEBUG**

**void DispDebugCode(void)**

**{**

**unsigned int i,j;**

**for(i=0,j=0;i<50;i++)**

**{**

**while(j<15)**

**{**

**if(debug\_code[i][j]==0xffff)**

**{**

**j=0;**

**break;**

**}**

**if(j==0)**

**{**

**if(debug\_code[i][j]==0)**

**{**

**i=50;**

**break;**

**}**

**else**

**{**

**WriteComm(debug\_code[i][j]);**

**}**

**}**

**else**

**{**

**WriteData(debug\_code[i][j]);**

**}**

**j++;**

**}**

**while(j<15)**

**{**

**if(debug\_code[i][j]==0xffff)**

**{**

**j=0;**

**break;**

**}**

**if(j==0)**

**{**

**if(debug\_code[i][j]==0)**

**{**

**i=50;**

**break;**

**}**

**else**

**{**

**WriteComm(debug\_code[i][j]);**

**}**

**DispStr("0X",0,(FONT\_H+1)\*(j+1),RED,WHITE);**

**DispInt(debug\_code[i][j],FONT\_W\*2,(FONT\_H+1)\*(j+1),RED,WHITE);**

**}**

**else**

**{**

**DispStr("0X",0,(FONT\_H+1)\*(j+1),BLUE,WHITE);**

**DispInt(debug\_code[i][j],FONT\_W\*2,(FONT\_H+1)\*(j+1),BLUE,WHITE);**

**}**

**j++;**

**}**

**WaitKey(); Delay(800);**

**}**

**}**

**void Debug(void)**

**{**

**DispStr("DEBUG CODE",0,0,BLACK,WHITE);**

**while(1)**

**{**

**DispDebugCode();**

**}**

**}**

**#endif**

**#ifdef DRAW\_FUNCTION**

**void PutPixel(unsigned int x,unsigned int y,unsigned int color)**

**{**

**BlockWrites(x,x+1,y,y+1);**

**CS0=0;**

**RD0=1;**

**RS=1;**

**DBH=color>>8;**

**DBL=color;**

**WR0=0;**

**WR0=1;**

**WR0=0;**

**WR0=1;**

**WR0=0;**

**WR0=1;**

**WR0=0;**

**WR0=1;**

**CS0=1;**

**}**

**unsigned int abs(unsigned int a,unsigned int b)**

**{**

**if(a>b)**

**return (a-b);**

**else**

**return (b-a);**

**}**

**void InterBresenhamline (int x0,int y0,int x1, int y1,int color)**

**{**

**int x, y, dx, dy,e,i;**

**dx = abs(x1,x0);**

**dy = abs(y1,y0);**

**if(dy<=dx)**

**{**

**e=-dx;**

**x=x0;**

**y=y0;**

**for (i=0; i<dx; i++){**

**PutPixel (x, y, color);**

**x++;**

**e=e+2\*dy;**

**if (e >= 0)**

**{**

**if(y1>y0)**

**y++;**

**else**

**y--;**

**e=e-2\*dx;**

**}**

**}**

**}**

**else**

**{**

**e=-dy;**

**x=x0;**

**if(y1>y0)**

**y=y0;**

**else**

**{**

**y=y1;**

**x=x1;**

**}**

**for (i=0; i<dy; i++){**

**PutPixel (x, y, color);**

**y++;**

**e=e+2\*dx;**

**if (e >= 0)**

**{**

**if(y1>y0)**

**x++;**

**else**

**x--;**

**e=e-2\*dy;**

**}**

**}**

**}**

**}**

**void DrawLine(unsigned int Xstart,unsigned int Xend,unsigned int Ystart,unsigned int Yend,unsigned int color)**

**{**

**unsigned int i,j;**

**BlockWrites(Xstart,Xend,Ystart,Yend);**

**for(i=Ystart;i<Yend+1;i++)**

**{**

**for(j=Xstart;j<Xend+1;j++)**

**{**

**WriteOneDot(color);**

**}**

**}**

**}**

**#endif**

**#ifdef DEBUG**

**unsigned int code debug\_code[50][15]=**

**{**

**{0xd1,0x00,0x1d,0x0c,END},**

**{0xd1,0x00,0x1d,0x08,END},**

**{0xd1,0x00,0x1d,0x07,END},**

**{0xd1,0x00,0x1d,0x0b,END},**

**{0},//end**

**};**

**#endif**

**//main.c**

**#include<reg52.h>**

**#include <intrins.h>**

**#include <absacc.h>**

**#include <stdlib.h>**

**#include "picture.h"**

**#include <main.h>**

**#include <lcd\lcd.h>**

**xdata unsigned char Packet[64];**

**unsigned char f\_down=0;**

**unsigned char f\_debug=0;**

**unsigned char f\_read=0;**

**unsigned char f\_para=0;**

**unsigned char f\_init=0;**

**unsigned char f\_other=0;**

**unsigned char pic=0;**

**unsigned int bf=0;**

**unsigned long int sd=1;**

**unsigned char VerifyCode0=0xaa;**

**unsigned char VerifyCode1=0x55;**

**unsigned char VerifyCode2=0xff;**

**unsigned char code tables[3][40]=**

**{**

**0x80,0x94,0xa7,0xb9,0xca,0xd9,0xe7,0xf1,0xf8,0xfd,**

**0xff,0xfd,0xf8,0xf1,0xe7,0xd9,0xca,0xb9,0xa7,0x94,**

**0x80,0x6c,0x55,0x46,0x35,0x25,0x18,0x0e,0x06,0x02,**

**0x00,0x02,0x06,0x0e,0x18,0x25,0x35,0x46,0x55,0x6c,**

**0x80,0x8c,0x99,0xa6,0xb3,0xc0,0xcc,0xd9,0xe6,0xf2,**

**0xff,0xf2,0xe6,0xd9,0xcc,0xc0,0xb3,0xa6,0x99,0x8c,**

**0x80,0x73,0x66,0x5a,0x4d,0x40,0x33,0x26,0x1a,0x0d,**

**0x00,0x0d,0x1a,0x26,0x33,0x40,0x4d,0x5a,0x66,0x73,**

**0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,**

**0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,0xff,**

**0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,**

**0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00**

**};**

**void DrawScop()**

**{**

**DrawLine(20,620,20,20,RED);**

**DrawLine(20,620,108,108,RED);**

**DrawLine(20,620,196,196,RED);**

**DrawLine(20,620,284,284,RED);**

**DrawLine(20,620,372,372,RED);**

**DrawLine(20,620,460,460,RED);**

**DrawLine(20,20,20,460,RED);**

**DrawLine(120,120,20,460,RED);**

**DrawLine(220,220,20,460,RED);**

**DrawLine(320,320,20,460,RED);**

**DrawLine(420,420,20,460,RED);**

**DrawLine(520,520,20,460,RED);**

**DrawLine(620,620,20,460,RED);**

**}**

**void main(void)**

**{**

**unsigned char i=0,n=0;**

**Devices\_Init();**

**DispColor(20,20,620,460,BLACK);**

**DrawScop();**

**while(1)**

**{**

**i++;**

**i%=29;**

**if(i==0)**

**{**

**n++;**

**n%=3;**

**}**

**DisSin(tables[n],40,i+1,GREEN);**

**Delay(8000);**

**DisSin(tables[n],40,i+1,BLACK);**

**DrawScop();**

**}**

**}**

**void Delay(unsigned int dly)**

**{**

**unsigned int i,j;**

**for(i=0;i<dly;i++)**

**for(j=0;j<2;j++);**

**}**

**void MCU\_Init(void)**

**{**

**P0= 0x0f; //数字输入(无模拟输入）**

**P1= 0xFF;**

**P2= 0xFF;**

**P3= 0xFF;**

**}**

**void Devices\_Init(void)**

**{**

**MCU\_Init();**

**Delay(10);**

**LCD\_Init();**

**DispColor(0,0,COL,ROW,GRAY50);**

**P0= 0xf0;**

**}**

**实验（二）**

**六、针对本实验写出实验心得**

**通过本次实验了解了单片机和LED的工作原理、I/O访问方式、单片机硬件制式，熟悉了Keil uVision4和STC-ISP软件的使用、程序编译和烧录方式。在烧录时应注意串口连线和对串口的保护，避免错误操作导致串口烧毁。**