单片机原理与应用

DANPIANJI YUANLI YU YINGYONG

单片机原理与应用

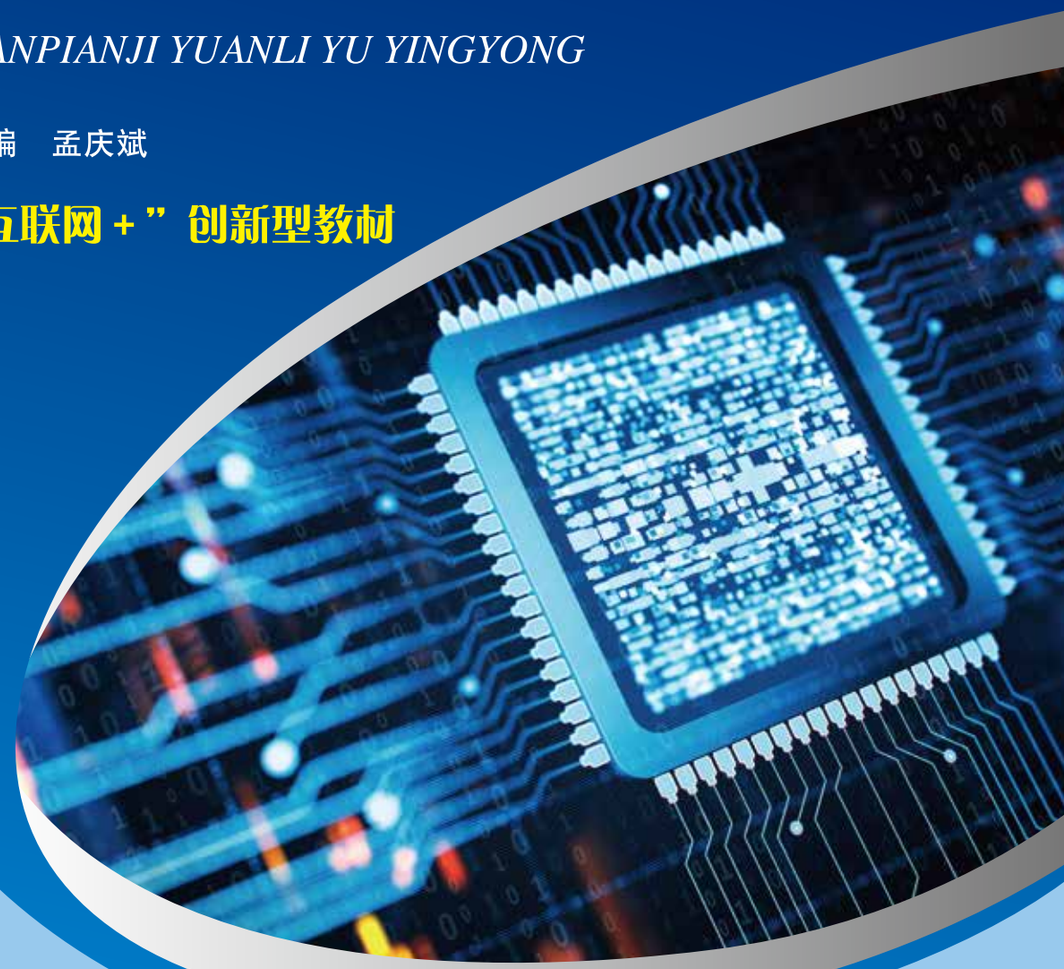
主编　孟庆斌

单片机原理与应用

DANPIANJI YUANLI YU YINGYONG

主编　孟庆斌

“互联网＋”创新型教材



ISBN 978-7-5608-9045-6
9 787560 890456 >
定价：49.80元

同济大学出版社
TONGJI UNIVERSITY PRESS

智慧学习平台

同济大学出版社
TONGJI UNIVERSITY PRESS

单片机原理与应用

DANPIANJI YUANLI YU YINGYONG

主　编　孟庆斌
副主编　高　艺　鞠　兰
　　赵二刚

同济大学出版社
TONGJI UNIVERSITY PRESS

内 容 提 要

本书共10章，主要内容包括初识单片机、多用途报警器的设计、制作一个计时的时钟、采集环境信息、让车轮转起来、蓝牙及其他通信方式、单片机组成原理、汇编语言基础、C51语言基础、智能室内环境监控系统设计。

本书可作为应用型本科院校电子信息、通信工程等专业的教材，也可作为相关技术人员的参考书。

图书在版编目(CIP)数据

单片机原理与应用 / 孟庆斌主编. --上海：同济大学出版社，2021.12

ISBN 978-7-5608-9045-6

Ⅰ.①单… Ⅱ.①孟… Ⅲ.①微控制器-教材 Ⅳ.①TP368.1

中国版本图书馆CIP数据核字(2021)第257027号

单片机原理与应用

孟庆斌 **主编**

责任编辑 张平官 **责任校对** 刘 睿 **封面设计** 刘文东

出版发行 同济大学出版社 www.tongjipress.com.cn

(地址：上海市四平路1239号 邮编：200092 电话：021-65985622)

经 销 全国各地新华书店

印 刷 大厂回族自治县聚鑫印刷有限责任公司

开 本 787 mm×1 092 mm 1/16

印 张 17插页1

字 数 352 000

版 次 2021年12月第1版 2021年12月第1次印刷

书 号 ISBN 978-7-5608-9045-6

定 价 49.80元

前　言 Preface

以8051系列为代表的8位单片机在各个领域得到了广泛的应用，尤其是在信息采集和工业控制领域发挥着巨大的作用。编者在高校从事单片机教学多年，对目前国内高校的课程情况和市场上的单片机教程有一定了解。单片机的理论比较艰涩，许多教材的理论教学与实验教学没有很好地融合和衔接，很多学生在学习理论知识的时候容易产生困惑和畏难情绪，课程授课效果不甚理想。通过多年的探索，编者认为，遵循从感性到理性的认知规律，将理论教学融于实践之中，边练边学边总结提高，有助于学生对理论知识的理解，以及将理论应用于实践。这就是编者编写本书的初衷。

编者在编写本书过程中遵循了如下思路。

(1)以有趣的实例为载体，通过具体可操作的步骤逐步介绍单片机最小系统、I/O接口、定时器/计数器、中断、串行口通信等，并在每一个实例中分别介绍相关的汇编指令和C51程序。

(2)在总结多个实例的基础上，集中、系统地介绍了51系列单片机的存储空间、指令时序、指令系统、汇编语言和C51程序语言。

(3)最后以一个综合单片机应用系统为对象，介绍项目方案设计、硬件系统构建和控制程序开发的整个过程。

本书遵循从感性到理性的认知规律，融理论教学于实践之中，不仅重视知识的传授，同时注重引导和培养学习者的项目开发思维，传授工程开发经验。即使是初学者，在本书的指导下，每完成一个模块的设计和实现，都会获得一定的成就感，增强对单片机学习的兴趣和乐趣。而随着学习的继续，看到亲手实现的综合性高、功能复杂的电子系统，学习者无论是在理论知识、动手能力、实践经验还是学习信心等方面，都将做好迎接更大挑战的准备。

本书由南开大学滨海学院孟庆斌任主编，南开大学高艺、鞠兰、赵二刚任副主编。具体编写分工如下：孟庆斌编写第 1 章至第 4 章，鞠兰编写第 5 章和第 6 章，赵二刚编写第 7 章和第 8 章，高艺编写第 9 章和第 10 章。本书在编写过程中，得到了南开大学滨海学院电子科学系和南开大学电子信息实验教学中心各位老师的支持与帮助，在此一并致以衷心的感谢。

本书中难免存在疏漏、不妥之处，敬请广大读者批评指正。

编　者

目　录 Contents

第 1 章

初识单片机

【学习目标】

(1)了解单片机及最小系统。

(2)学习 Keil 软件开发环境。

(3)学习 Proteus 仿真环境。

1.1 彩灯控制实例

单片机,即微控制单元(microcontroller unit,MCU),采用超大规模集成电路技术把具有数据处理能力的中央处理器(CPU)、随机存取存储器(random access memory,RAM)、只读存储器(read-only memory,ROM)、输入/输出接口(I/O 接口),可能还包括定时器/计数器、串行通信接口、显示驱动电路(LCD 或 LED 驱动电路)、脉宽调制(pulse width modulation,PWM)电路、模拟多路转换器及模/数(A/D)转换器等模块,集成到一块芯片上,构成一个小而完善的计算机系统。

单片机能够单独完成现代工业控制所需的智能化控制功能,这是单片机的一大应用。单片机的出现,使控制系统实现了智能化,是控制技术的一次革命和重要的里程碑。现在,单片机广泛应用于智能仪器仪表、工业控制、家用电器、计算机网络和通信、医用设备等领域。

1. 在智能仪器仪表上的应用

单片机具有体积小、功耗低、控制功能强、扩展灵活、微型化和使用方便等优点,结合不同类型的传感器,可实现电压、功率、频率、湿度、温度、流量、速度、厚度、角度、长度、硬度、元素、压力等的测量。采用单片机控制使得仪器仪表数字化、智能化、微型化,且功能相比单纯采用模拟或数字电路更强大,特别适用于精密的测量设备,如功率计、示波器、各种分析仪等。

2. 在工业控制中的应用

单片机可以用来构成形式多样的控制系统、数据采集系统,如工厂流水线的智能化管理、电梯智能化控制、各种报警系统,以及与计算机联网构成的二级控制系统等。

3. 在家用电器中的应用

现在的家用电器基本上都采用了单片机控制,如电饭煲、洗衣机、电冰箱、空调、彩电等。

4. 在计算机网络和通信领域中的应用

当今的通信设备基本上都实现了单片机智能控制,无论是手机、电话、小型程控交换机、楼宇自动通信呼叫系统、列车无线通信系统,还是日常工作中随处可见的集群移动通信、无

线电对讲机等。

5. 在医用设备中的应用

单片机在医用设备中的用途也相当广泛，如医用呼吸机、各种分析仪、监护仪、超声诊断设备及病床呼叫系统等。

此外，单片机在工商、金融、科研、教育、国防、航空航天等领域也有着十分广泛的用途。

在市场上，单片机品种繁多，各具特色，主要产品有 MCS-51 系列、Motorola 公司的 68 系列、Microchip 公司的 PIC16F/18F 系列等。MCS-51 是对所有兼容 Intel 8031 指令系统的单片机的统称，起源于 Intel 的 8031 单片机。后来随着 Flash Memory 技术的发展，其成为应用最广泛的 8 位单片机之一，其代表型号是原 Atmel 公司的 AT89 系列。Intel 公司将 MCS-51 的核心技术授权给其他公司，很多公司为满足不同的需求推出了功能略有不同于 MCS-51 系列的兼容机型。目前主要的生产商及型号如下。

(1)Intel：80C31、80C51、87C51，80C32、80C52、87C52 等。

(2)Microchip：89C51、89C52、89C2051、89S51、89S52 等。

(3)Philips、华邦、Dallas、Siemens(Infineon)等公司的许多产品。

(4)国产宏晶 STC 单片机。

其中，STC 单片机以其低功耗、廉价、稳定的性能，占据着国内 51 单片机的较大市场。

接下来，通过一个彩灯控制的实例初步认识单片机。图 1-1-1 所示为彩灯控制电路，电路中采用 STC12C5A60S2 单片机控制 3 个彩色 LED 依次闪烁，闪烁时间间隔为 500 ms。

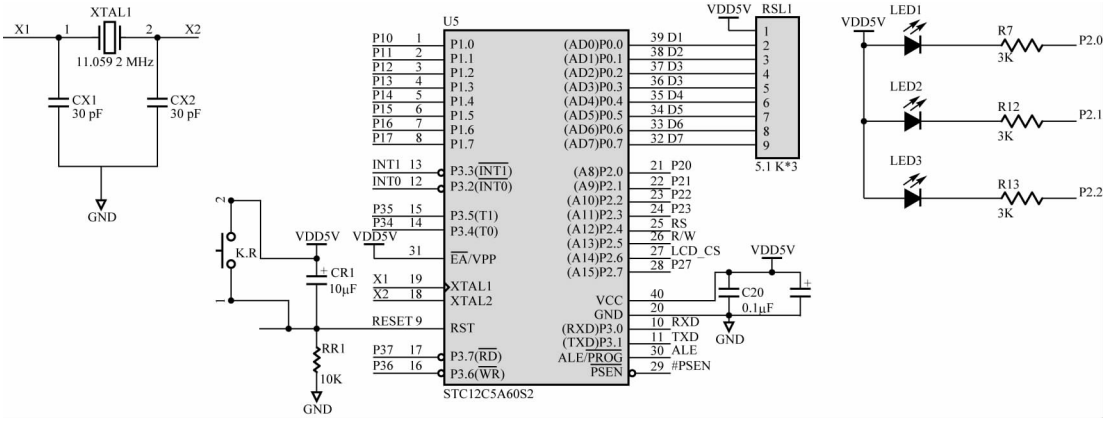


图 1-1-1　彩灯控制电路

观察电路，LED1、LED2、LED3 的负极分别经限流电阻接到 P2.0、P2.1、P2.2 这 3 个引脚上，正极都接到 5 V 电源上，即当单片机 P2.0、P2.1、P2.2 引脚输出低电平时，LED1、LED2、LED3 点亮。因此，对 3 个 LED 的亮、灭控制需要通过在相应的引脚输出低、高电平来实现，即编写单片机控制程序使 P2.0、P2.1、P2.2 引脚依次输出持续 500 ms 的低电平，实现 3 个 LED 依次闪烁，且闪烁时间间隔为 500 ms。

根据上述分析，可以得到彩灯控制工作流程图，如图 1-1-2 所示。

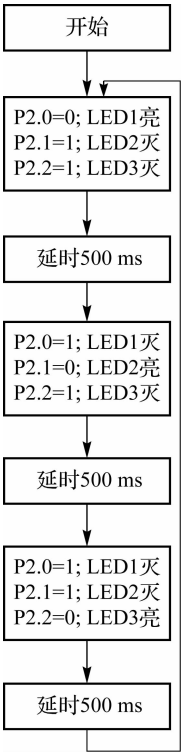


图 1-1-2　彩灯控制工作流程图

首先看点亮 LED 的控制程序，分别用汇编语言和 C51 语言来实现。以点亮 LED1 为例，汇编程序代码如下。

```
        ;功能:点亮 LED1
        ORG   0000H
        AJMP   MAIN
        ORG   0030H
MAIN:   CLR   P2.0              ;LED1 亮
        END
```

C51 程序代码如下。

方法一：

```
    /*------------------------------------------------------------
                功能:点亮 LED1
    ------------------------------------------------------------*/
    #include<reg52.h>      //包含特殊功能寄存器定义的头文件
    sbit LED1=P2^0;         //用 sbit 关键字定义 LED1 连接到 P2.0 端口
    /*------------------------------------------------------------
                    主函数
    ------------------------------------------------------------*/
void main (void)        //任何一个 C 程序都必须有且只有一个 main 函数
{
```

```
        LED1＝1;                //将 P2.0 口赋值 1,输出高电平
        LED1＝0;                //将 P2.0 口赋值 0,输出低电平
        while (1)
        {
                                //其中加入其他需要一直工作的程序
        }
        }
```

方法二：

```
        /*————————————————————————————————————————
                功能:点亮 LED1
        ————————————————————————————————————————*/
        #include<reg52.h>      //包含特殊功能寄存器定义的头文件
        /*————————————————————————————————————————
                主函数
        ————————————————————————————————————————*/
        void main (void)
        {
        P2 = 0xFF;              //P2 口全部为高电平
        P2 = 0xFE;              //P2 口的最低位点亮
        while (1)
        {
                                //其中加入其他需要一直工作的程序
        }
        }
```

然后加入延时程序,实现对 3 个 LED 的依次闪烁控制,汇编程序代码如下。

```
        ;功能:3 个 LED 依次闪烁
        ORG     0000H
        AJMP    MAIN
        ORG     0030H
MAIN:   CLR     P2.0            ;LED1 亮
        SETB    P2.1            ;LED2 灭
        SETB    P2.2            ;LED3 灭
        ACALL   DELAY500MS      ;延时 500 ms
        SETB    P2.0            ;LED1 灭
        CLR     P2.1            ;LED2 亮
        SETB    P2.2            ;LED3 灭
        ACALL   DELAY500MS      ;延时 500 ms
        SETB    P2.0            ;LED1 灭
        SETB    P2.1            ;LED2 灭
        CLR     P2.2            ;LED3 亮
```

```
        ACALL   DELAY500MS          ;延时 500 ms
        AJMP    MAIN
DELAY500MS:                         ;@11.059 2 MHz,延时 500 ms
        MOV     30H,#17
        MOV     31H,#208
        MOV     32H,#23
    NEXT:
        DJNZ    32H,NEXT
        DJNZ    31H,NEXT
        DJNZ    30H,NEXT
        RET
        END
```

C51 程序代码如下。

```
/*————————————————————————————————————
功能:3 个 LED 依次闪烁
————————————————————————————————————*/
#include<reg52.h>
sbit LED1=P2^0;
sbit LED2=P2^1;
sbit LED3=P2^2;
void Delay(unsigned int t);         //函数声明
/*————————————————————————————————————
                        主函数
————————————————————————————————————*/
void main (void)
{
LED1=1;                 //赋初始值
LED2=1;
LED3=1;
while (1)
{
    LED1=0;             //LED1 亮
    LED2=1;             //LED2 灭
    LED3=1;             //LED3 灭
    Delay(50000);
    LED1=1;             //LED1 灭
    LED2=0;             //LED2 亮
    LED3=1;             //LED3 灭
    Delay(50000);
    LED1=1;             //LED1 灭
```

```
        LED2=1;                 //LED2 灭
        LED3=0;                 //LED3 亮
        Delay(50000);
        }
}
/*------------------------------------------------
延时函数,含有输入参数 unsigned int t,无返回值
unsigned int 定义无符号整型变量,其值的范围是 0~65 535
------------------------------------------------*/
void Delay(unsigned int t)
{
while(--t);
}
```

将上述程序代码编译成 51 单片机可执行的机器码,通过下载工具下载到单片机内部的程序存储空间。当电路上电后,单片机即可按照程序执行,控制 LED1、LED2、LED3 依次闪烁,闪烁时间间隔为 500 ms。

通过彩灯控制的例子可以看到,构建单片机应用系统,需要设计单片机的外围电路和接口电路,这涉及单片机的硬件开发环境;还需要设计单片机的控制程序,这涉及单片机的软件开发环境。接下来将分别对硬件开发环境和软件开发环境进行介绍。

1.2 单片机最小系统

单片机及能够使其工作的最小外围电路称为单片机的最小系统,包括系统供电、复位控制和时钟三部分外围电路。

1.2.1 系统供电电路

视频
单片机最小系统

对于任何一个电子系统来讲,首要问题都是为整个系统供电,供电电路的稳定可靠是系统平稳运行的前提和基础。有些单片机应用系统在受到外界干扰时会出现程序“跑飞”的现象,而避免出现这种现象的一个重要手段就是为单片机系统配置一个稳定可靠的电源供电模块。

为提高使用的便捷性,很多单片机开发板采用 USB 接口供电和外部稳定电源供电两种供电形式共存的供电电路。目前,主流单片机及常用外设芯片的供电电源分为 5 V 和 3.3 V 两种标准。本书所采用的 STC89C52 单片机使用 5 V 供电标准。为适应一些外设芯片和兼容型单片机对 3.3 V 供电标准的要求,本书所采用的系统供电电路既提供 5 V 输出,也提供3.3 V 输出。系统供电电路如图 1-2-1 所示。

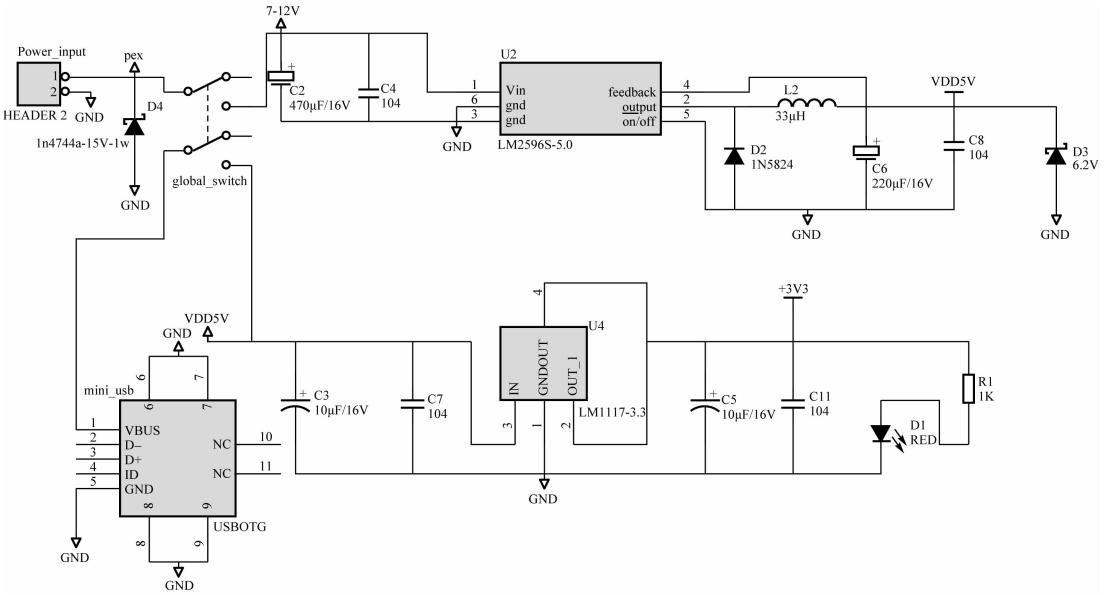


图 1-2-1　系统供电电路

40 脚封装的单片机，电源引脚是 40 脚和 20 脚：40 脚接＋5 V，通常称为 VCC 或 VDD，代表的是电源正极；20 脚接参考地，通常称为 GND 或 VSS，代表的是电源的负极。VCC 和 GND 之间一般接入一个滤波电容，用于滤除高频干扰。

1.2.2　复位控制电路

在单片机系统中，复位控制电路是非常关键的。单片机复位类似于计算机的重启。当计算机在使用中出现死机等意外情况时，按下重启按钮，计算机内部程序从头开始重新执行。单片机也一样，当单片机系统在运行中出现程序“跑飞”等意外情况时，按下复位按钮，控制程序也自动从头开始重新执行。另外，当单片机系统重新上电时，控制程序也应自动从头开始重新执行。复位控制电路可由电容、电阻的串联电路构成，如图 1-2-2 所示。

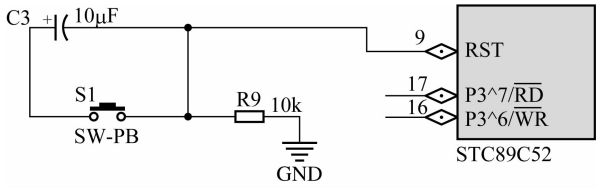


图 1-2-2　复位控制电路

观察电路，结合“电容电压不能突变”的性质，可以知道，当系统上电时，RST 脚会出现高电平，并且这个高电平持续的时间由电路的 R、C 值决定。对于典型的 51 单片机，RST 脚的高电平持续两个机器周期以上就可将单片机有效复位。所以，适当组合 R、C 的取值可以保证可靠复位，其典型值为 10 kΩ 和 10 μF。在单片机系统中，系统上电启动的时候复位一次，称为上电复位。在系统运行过程中，如果按下复位按钮，电容迅速放电。当按钮释放后，RST 脚被拉到高电平，从而将单片机复位。通过复位按钮，使用者可以随时对单片机进行复位控制。

1.2.3 时钟电路

时钟电路为单片机系统提供基准时钟信号，单片机内部所有电路都是以这个时钟信号为步调基准进行工作的。STC89C52 单片机的 XTAL1 脚和 XTAL2 脚是时钟引脚。单片机可以采用外部的独立时钟信号作为系统时钟，此时时钟信号通过 XTAL1 脚接入单片机内部。但一般利用单片机内部集成的时钟产生电路来获得系统时钟，这时需要在 XTAL1 脚和 XTAL2 脚之间接一个由晶体振荡器（晶振）和电容构成的电路。

常用的单片机开发板的系统时钟频率为 11.059 2 MHz 或 12 MHz。这里采用 11.059 2 MHz 时钟，其电路如图 1-2-3 所示。电路中采用的是 11.059 2 MHz 晶振，两个电容均为 33 pF。

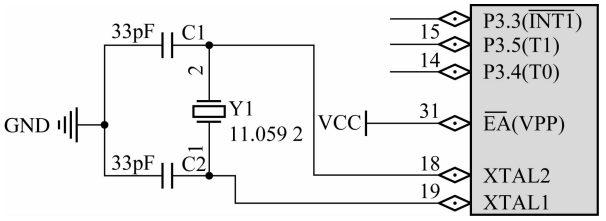


图 1-2-3 时钟电路

1.2.4 开发板的其他资源和程序下载

单片机在与系统供电电路、复位控制电路和时钟电路构成最小系统后，即可运行控制程序，但还不能形成应用。单片机应用系统在最小系统之上还要附加必要的外围电路。常用的单片机开发板一般包括按键、LED、LCD、蜂鸣器、串口等外围电路。如前述的彩灯控制系统，其外围电路为 3 个 LED 及其驱动电路。本书后续章节会陆续介绍一些外围接口电路及其应用，此处不再详述。

STC89C52 单片机可以采用串口进行程序下载，这为程序调试提供了巨大的便利，接下来将详细介绍程序下载的方法和流程。由于程序下载要用到串口，所以必须设计串口接口电路，如图 1-2-4 所示。

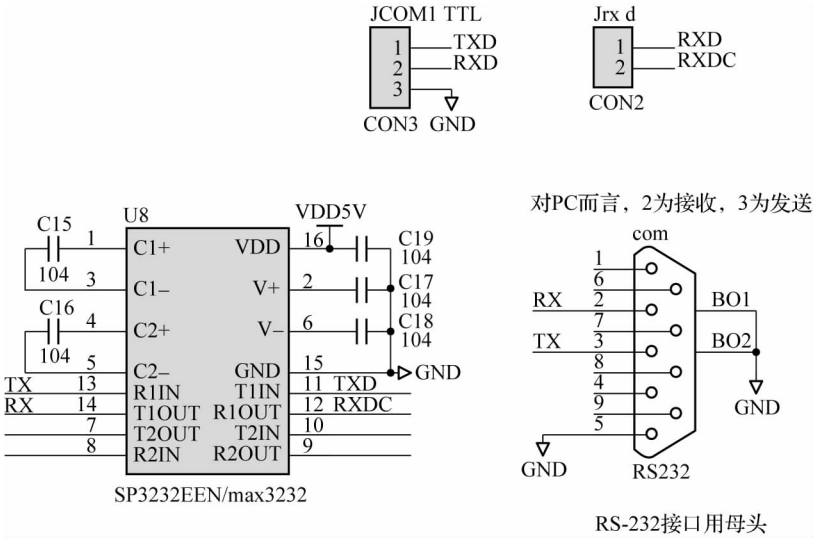


图 1-2-4 串口接口电路

图 1-2-4 中，串口转换芯片 max3232 实现单片机串口 TTL 电平与计算机串口电平之间的转换，使得计算机串口能够通过 RS-232 接口与单片机进行串行通信。

在对单片机进行程序下载时，首先检查单片机应用系统的硬件连接，保证接线准确无误；然后通过串口线将单片机开发板与计算机相连。程序下载是将程序编译软件编译生成的机器码文件（即. HEX 文件）下载到单片机的程序存储空间。程序下载要用到厂家提供的 STC-ISP 软件，程序下载过程如下。

（1）打开 STC-ISP 软件，出现图 1-2-5 所示界面。

图 1-2-5　打开 STC-ISP 软件界面

（2）选择单片机型号，在“单片机型号”下拉列表中选择自己使用的 MCU 类型，这里以选择 STC89C/LE52RC 为例，如图 1-2-6 所示。

图 1-2-6　选择单片机型号

注意，当选择过一次单片机型号后，软件会把选择的型号记录为“常用型号”，出现在最上面，方便下次选择，如图 1-2-7 所示。

图 1-2-7　显示常用型号

(3)选择单片机型号后，需要选择 COM 口。在“串口号”下拉列表中选择单片机与计算机连接的串口，新版 STC-ISP 软件在检测到后会自动选择，并显示连接的设备名称，如图 1-2-8 所示。

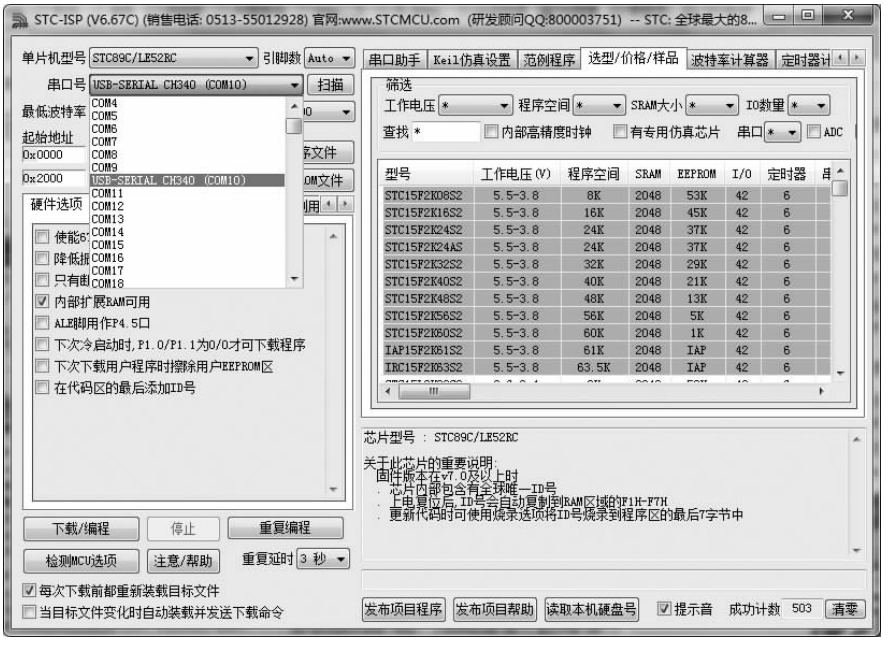


图 1-2-8　选择 COM 口

注意，如果没有自动显示已连接的串口，可能是下载器驱动没装好，或者 STC-ISP 软件版本较旧，这时需要手动查询串口号。打开“计算机管理”界面，选择“设备管理器”，如图 1-2-9 所示。

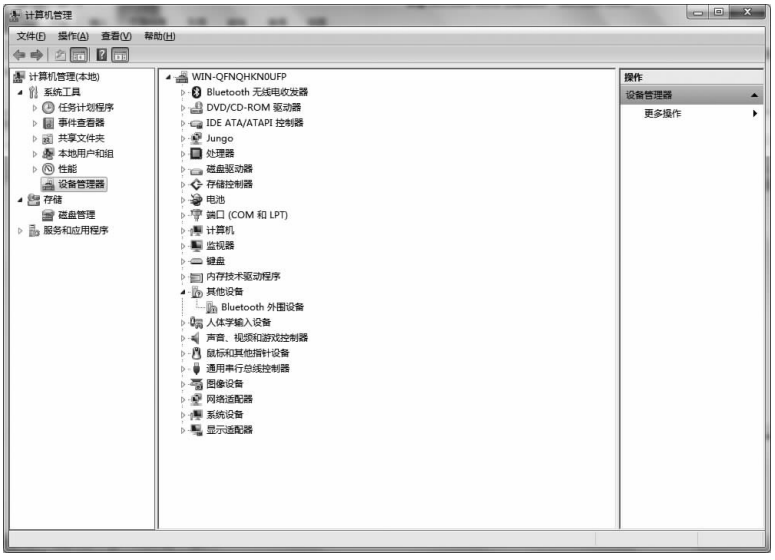


图 1-2-9　打开“设备管理器”

在“端口”列表下找到要使用的串口设备。如图 1-2-10 所示，这里使用的串口设备为 COM10。需要指出的是，如果没有相应的串口设备，说明下载器连接有问题；如果设备图标上有叹号，说明下载器的驱动没有装好。

图 1-2-10　显示 COM 口

(4)回到刚才选择串口的界面，选择完串口后，单击“打开程序文件”按钮，选择想要下载的程序，如图 1-2-11 所示。

图 1-2-11　选择想要下载的程序

(5)单击“下载/编程”按钮,下载程序文件。要注意,必须先单击“下载/编程”按钮,再给单片机上电,如图 1-2-12 所示。

图 1-2-12　下载程序文件

(6)下载成功会出现图 1-2-13 所示界面。

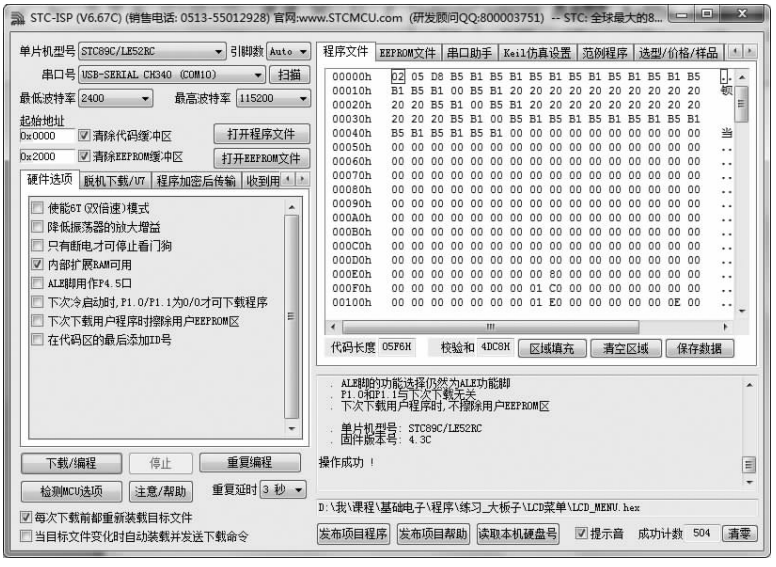


图 1-2-13　下载成功界面

1.3　Keil 软件开发环境

ARM Keil 软件开发环境是 ARM 公司提供的微控制器嵌入式软件开发环境,包括 MDK-ARM、C51、C251、C166 等开发工具,分别用于 Cortex 和 ARM 器件、8051、80251、C166 等微控制器的嵌入式软件开发。

Keil C51 开发工具支持所有 8051 衍生产品，提供了包括工业标准 Keil C 编译器、宏汇编器、调试器、库管理器、实时操作系统内核和一个功能强大的仿真器等在内的完整开发工具，并通过 Keil μVision4 IDE 集成环境将这些工具组合在一起。

Keil μVision4 IDE 集成环境中的工业标准 Keil C 编译器为 8051 的软件开发提供了 C 语言环境，同时保留了汇编程序高效、快速的特点。在该集成环境中可以准确地模拟 MCS-51 设备的片上外围设备（I^2C、CAN、UART、SPI、中断、I/O 端口、A/D 转换器、D/A 转换器和 PWM 模块），使开发者可以在没有目标设备的情况下使用集成环境编写和测试应用程序。

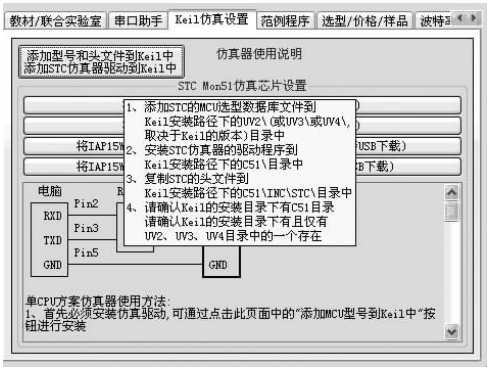
以前述彩灯控制为例，介绍在 Keil μVision4 IDE 集成环境下如何进行程序设计。

1. STC 设备库安装

在编辑、编译 STC 系列单片机应用程序时，可选用任何厂家的 51 或 52 系列单片机，再用汇编语言或 C 语言对 STC 系列单片机的新增特殊功能寄存器进行定义。也可以通过 STC-ISP 下载编程工具将 STC 型号 MCU 添加到 Keil μVision4 IDE 的设备库中。

如果需要在 Keil μVision4 IDE 的设备库中增加 STC 型号 MCU，则可按如下步骤进行设置。

（1）打开 STC-ISP 下载编程工具的最新软件 STC-ISP-V6.85，选择“Keil 仿真设置”选项卡，单击“添加型号和头文件到 Keil 中”按钮，如图 1-3-1 所示。



视频
Keil C51 软件的使用

图 1-3-1　将 STC 型号 MCU 添加到 Keil 的设备库中

（2）在弹出的“浏览文件夹”对话框中选择 Keil 安装目录（一般为“C:\Keil”），然后单击“确定”按钮，这样就将 STC 型号的 MCU 成功添加到 Keil 设备库中，如图 1-3-2 所示。

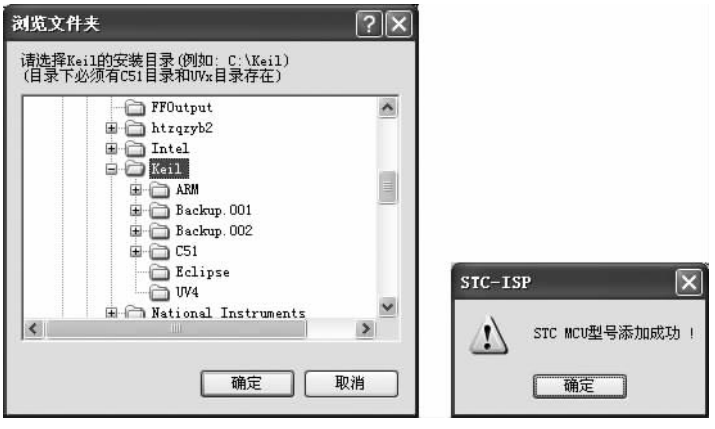


图 1-3-2　成功添加 STC 型号的 MCU 到 Keil 设备库中

2. 创建工程

下面详细介绍如何使用 Keil μVision4 IDE 开发、编译、调试用户程序。

(1)启动 Keil μVision4 IDE,进入 Keil μVision4 IDE 的主界面,如图 1-3-3 所示。

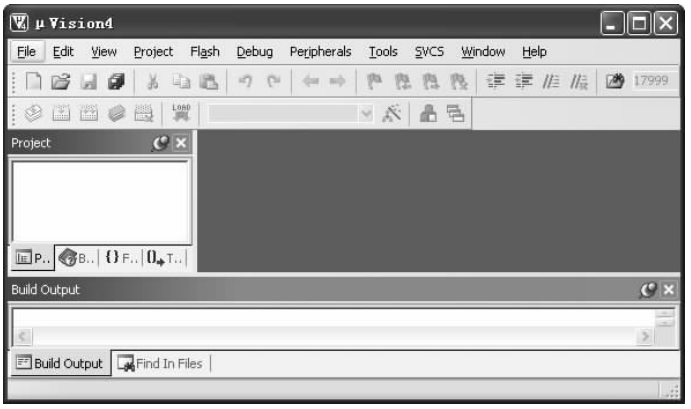


图 1-3-3 Keil μVision4 IDE 的主界面

(2)建立一个新工程。执行“Project”→“New μVision Project”命令,如图 1-3-4 所示。

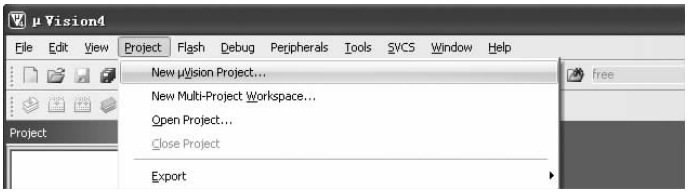


图 1-3-4 建立新工程

(3)在弹出的对话框中选择新项目要保存的路径和文件名,如保存路径为 E:\C51\LED,文件名为 LED,单击“保存”按钮即可。Keil μVision4 的项目文件扩展名为.uvproj,如图 1-3-5 所示。

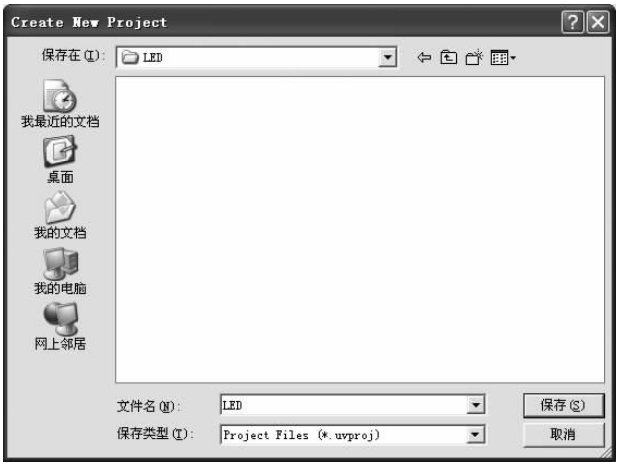


图 1-3-5 选择新项目保存的路径和文件名

(4)因为之前已经通过 STC-ISP 下载编程工具将 STC 型号 MCU 添加到 Keil μVision4 的设备库中,所以在上一步“保存”之后会弹出“Select a CPU Data Base File”(选择 CPU 数据库)对话框,如图 1-3-6 所示。该对话框中有“Generic CPU Data Base”(通用 CPU 数据库)和“STC MCU Database”(STC MCU 数据库)两个选项。如果用户所使用的单片机是 STC 系列,则选择“STC MCU Database”,然后单击“OK”按钮即可。

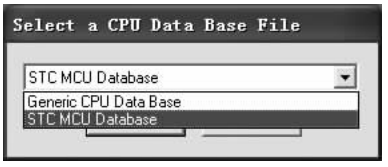


图 1-3-6　选择设备数据库

(5)选择设备数据库后会弹出“Select Device for Target”(选择所需器件)对话框,如图 1-3-7 所示。因为选择了“STC MCU Database”,所以这里的 MCU 型号都是 STC 型号,用户可在左侧的数据列表(Data base)中选择所使用的具体单片机型号。

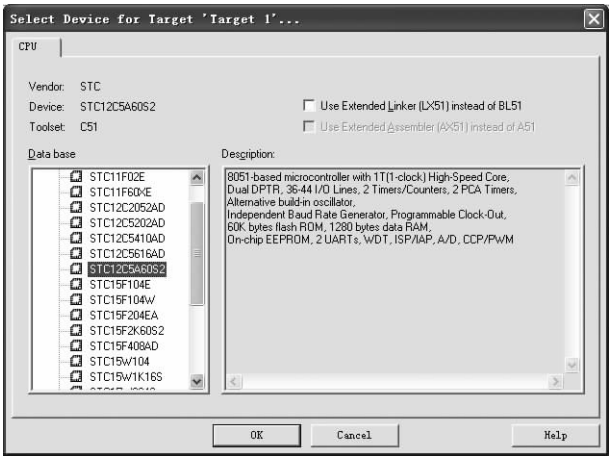


图 1-3-7　选择所需器件

(6)选择好单片机型号并单击“OK”按钮后,会弹出图 1-3-8 所示的对话框,询问是否将标准 51 初始化程序(STARTUP.A51)加入项目。单击“是”按钮,程序会自动复制标准 51 初始化程序到项目所在目录并将其加入项目。一般情况下,单击“否”按钮。

图 1-3-8　询问是否将标准 51 初始化程序加入项目

(7)项目建好后就可以开始编写程序。如图 1-3-9 所示,执行“File”→“New”命令,新建文件后的界面如图 1-3-10 所示。

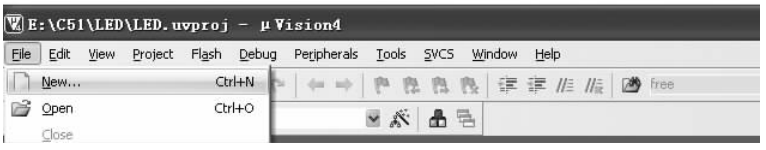


图 1-3-9　在项目下新建文件

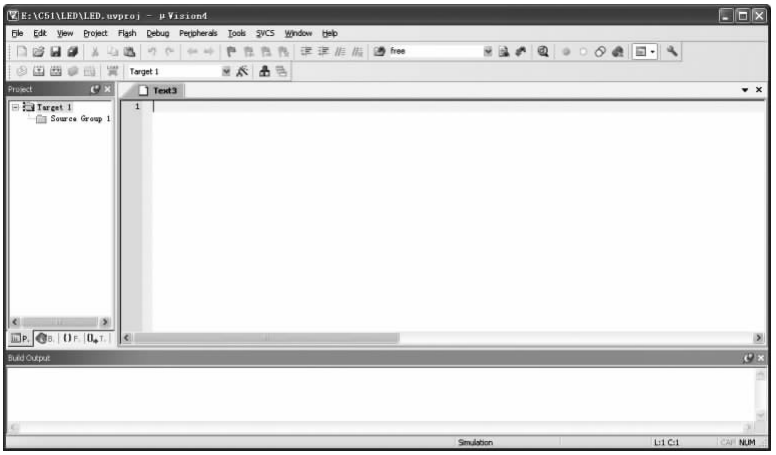


图 1-3-10　新建文件后的界面

此时光标在编辑窗口中闪烁，这时可以输入用户编写的应用程序。输入程序后执行“File”→“Save As”命令，系统会弹出图 1-3-11 所示界面。在“文件名”文本框中输入文件名，同时必须输入正确的扩展名。如果用 C 语言编写程序，则扩展名为. C；如果用汇编语言编写程序，则扩展名为. ASM。扩展名不分大小写。然后单击“保存”按钮。

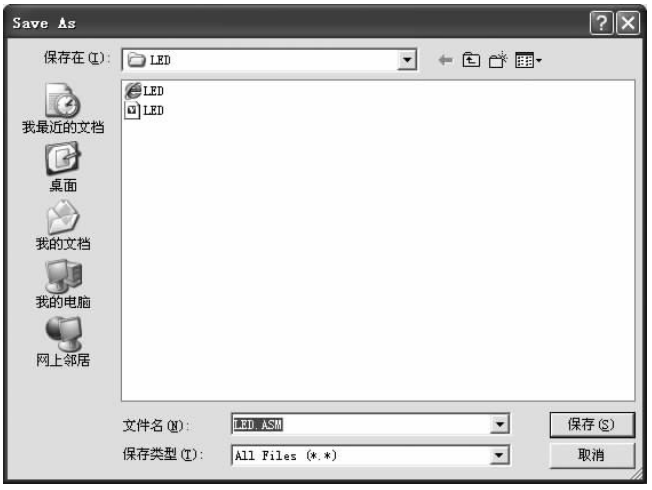


图 1-3-11　保存文件界面

(8)将应用程序添加到工程中。单击“Target 1”前面的“＋”号，展开后出现“Source Group 1”文件夹。右击该文件夹，在弹出的快捷菜单中选择“Add Files to Group 'Source Group 1'”命令，如图 1-3-12 所示，弹出添加文件对话框，将文件类型选择为“Asm Source file”，如图 1-3-13 所示。选中文件“LED. asm”，单击“Add”按钮，将该文件添加至工程。

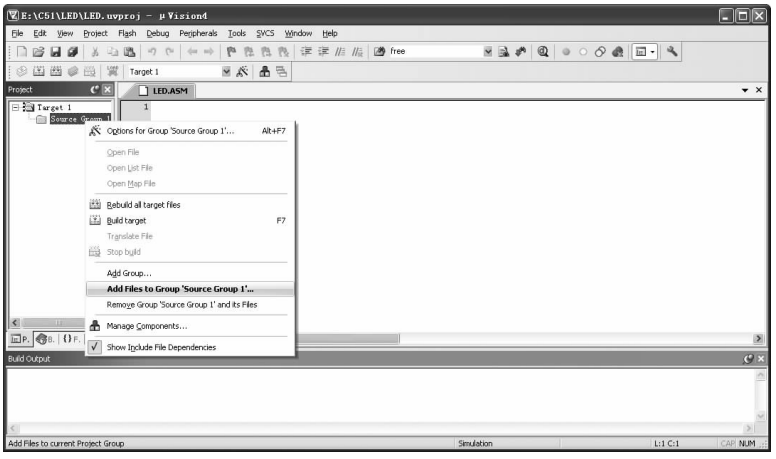


图 1-3-12　弹出快捷菜单界面

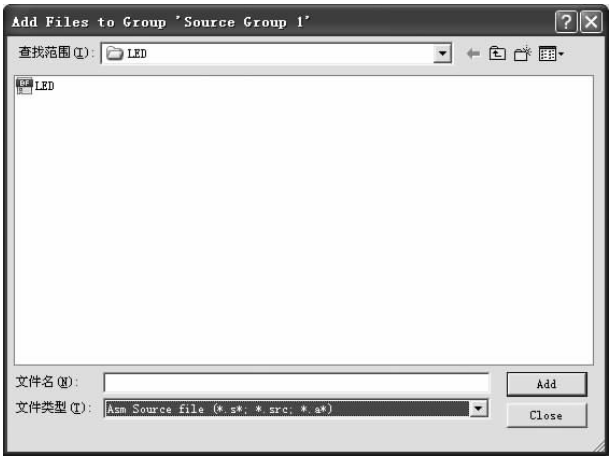


图 1-3-13　将文件添加至工程界面

(9)环境设置。如图 1-3-14 所示,在“Target 1”上右击,选择“Options for Target 'Target1'”命令,或执行“Project”→“Options for Target 'Target 1'”命令,系统会弹出“Options for Target 'Target 1'”对话框,如图 1-3-15 所示,在该对话框中设定目标的硬件环境。

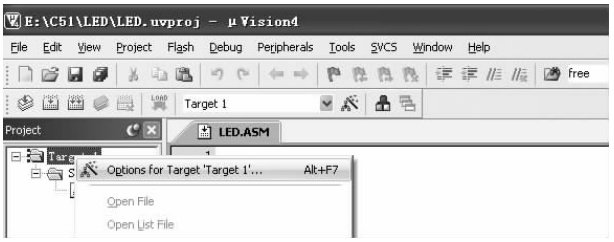


图 1-3-14　选择“Options for Target 'Target 1'”命令

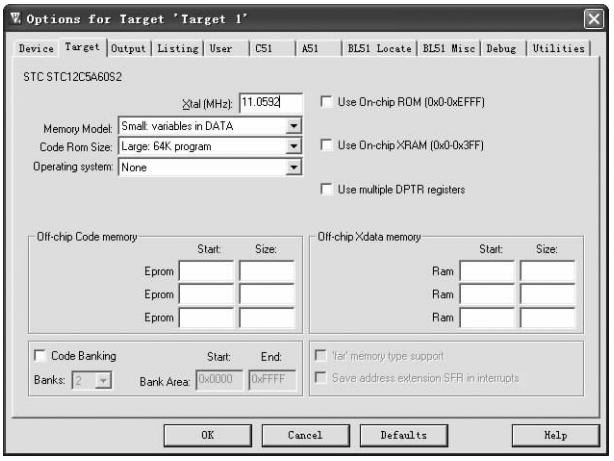


图 1-3-15　“Options for Target 'Target 1'”对话框

“Options for Target 'Target 1'”对话框中有多个选项卡，用于设备(Device)选择、目标(Target)属性、输出(Output)属性、C51 编译器属性、A51 编译器属性、BL51 连接器属性、调试(Debug)属性等信息的设置，一般情况下按默认设置即可。

需要注意的是，一定要设置在编译、连接程序时自动生成机器代码文件(. HEX)，因为默认是不输出 HEX 代码的，所以需要手动设置。单击“Output”选项卡，选中“Create HEX File”复选框，如图 1-3-16 所示，使程序编译后生成 HEX 代码文件(默认文件名为项目文件名，也可以在“Name of Executable”文本框中输入 HEX 文件的文件名)，单击“OK”按钮结束设置。

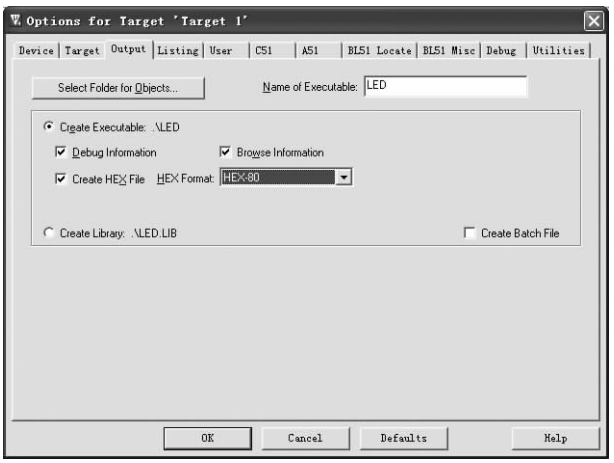


图 1-3-16　设置在编译、连接程序时自动生成机器代码文件(. HEX)

3. 代码输入

创建完工程以后，可以进行软件编程。在 LED. asm 文件中输入以下代码：

```
ORG     0000H
AJMP    MAIN
ORG     0030H
```

```
MAIN:   CLR     P2.0            ;LED1 亮
        SETB    P2.1            ;LED2 灭
        SETB    P2.2            ;LED3 灭
        ACALL   DELAY500MS      ;延时 500 ms
        SETB    P2.0            ;LED1 灭
        CLR     P2.1            ;LED2 亮
        SETB    P2.2            ;LED3 灭
        ACALL   DELAY500MS      ;延时 500 ms
        SETB    P2.0            ;LED1 灭
        SETB    P2.1            ;LED2 灭
        CLR     P2.2            ;LED3 亮
        ACALL   DELAY500MS      ;延时 500 ms
        AJMP    MAIN
DELAY500MS:                     ;@11.059 2 MHz,延时 500 ms
        MOV   30H,#17
        MOV   31H,#208
        MOV   32H,#23
NEXT:   DJNZ   32H,NEXT
        DJNZ   31H,NEXT
        DJNZ   30H,NEXT
        RET
        END
```

4. 程序编译与调试

程序编写完成后,需要进行编译与调试。

(1)程序编译。执行“Project”→“Rebuild all target files”命令,如图 1-3-17 所示。

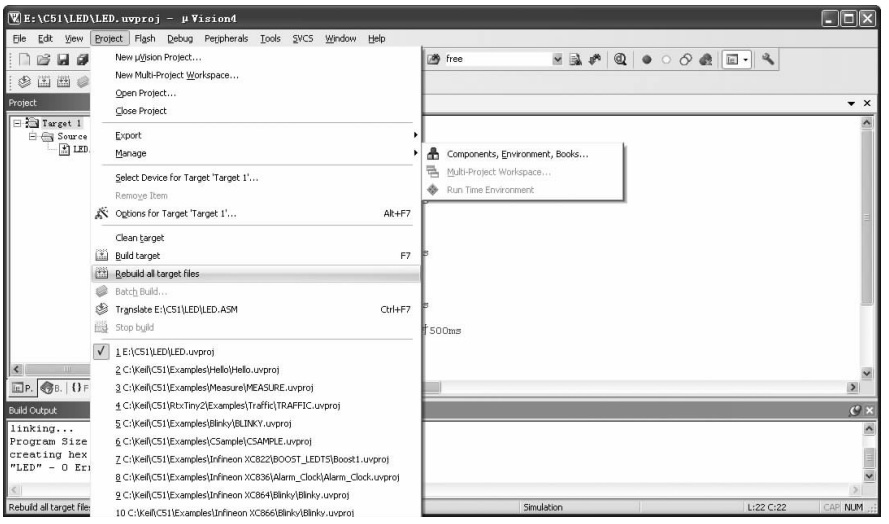


图 1-3-17　执行“Project”→“Rebuild all target files”命令

当输出显示信息为 0 错误、0 警告时，说明程序编译通过，可以进入程序调试，如图 1-3-18 所示。

图 1-3-18　“Build Output”输出信息界面

(2)程序调试。执行“Debug”→“Start/Stop Debug Session”命令，进入调试模式，界面如图 1-3-19 所示。

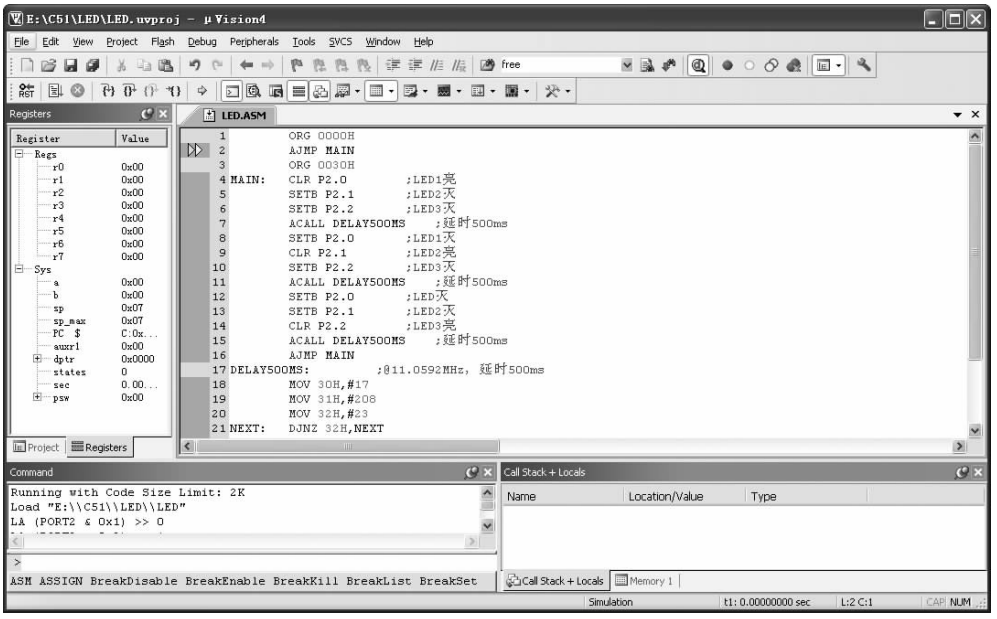


图 1-3-19　程序调试界面

(3)“Peripherals”菜单下有多种选项：“Interrupt”选项可以观察中断系统状态，“I/O-Ports”选项可以观察 I/O 端口的状态，“Serial”选项可以观察串口的工作状态，“Timer”选项可以观察定时器的工作状态，“Clock Control”选项可以观察时钟控制的信息。在本例中，主要观察 I/O 端口中 P2 端口的状态信息，则选择“I/O-Ports”→“Port 2”选项，如图 1-3-20 所示。

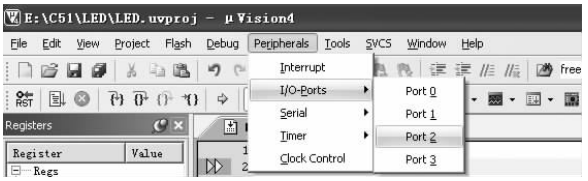


图 1-3-20　在“Peripherals”菜单中选择 P2 端口

此时弹出 P2 端口的观察窗口“Parallel Port 2”，如图 1-3-21 所示。窗口中，对钩表示高电平，空白表示低电平。

(4)单步调试。单击“Debug”菜单,弹出下拉菜单,如图 1-3-22 所示。

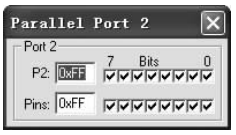


图 1-3-21　观察窗口“Parallel Port 2”

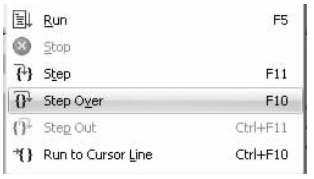


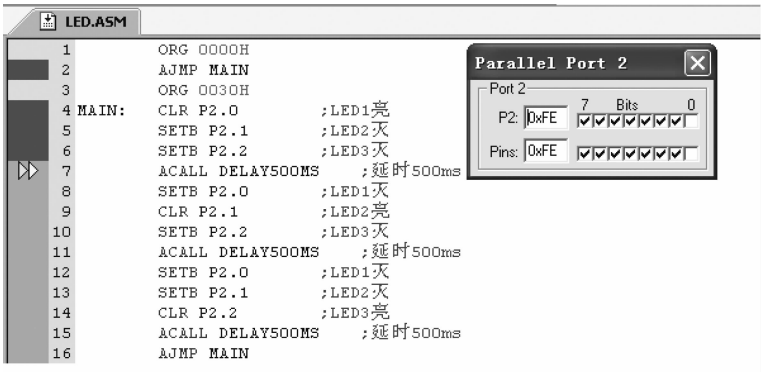
图 1-3-22　“Debug”下拉菜单

“Debug”下拉菜单中每一项指令的功能如表 1-3-1 所示。

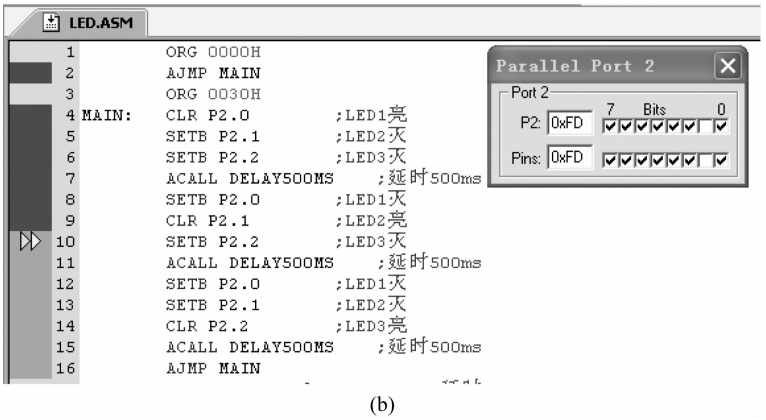
表 1-3-1　“Debug”下拉菜单下的指令功能

指　令	功　能
Run	全速运行
Stop	停止
Step	单步执行,深入子函数内部
Step Over	单步执行,将子函数视为一条指令
Step Out	将子函数运行完后跳出子函数
Run to Cursor Line	运行到光标行

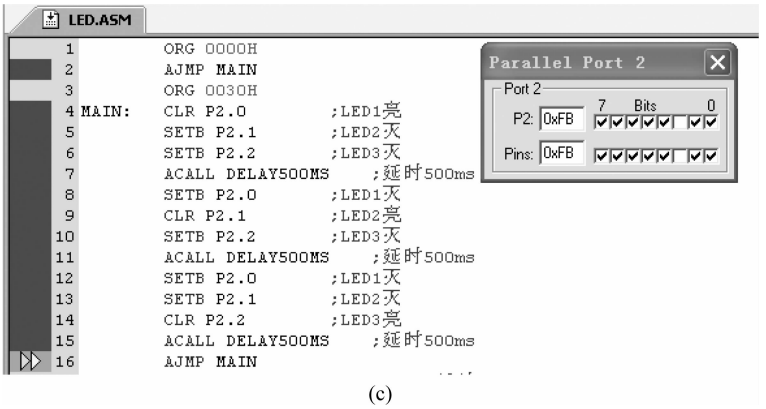
本例中,选择“Step Over”选项进行单步调试,可以观察到 P2 口的不同状态,如图 1-3-23 所示。



(a)



(b)



(c)

图 1-3-23　对 LED 程序进行单步调试时得到 P2 端口的不同显示结果

(5)虚拟逻辑分析仪的使用。利用 Keil 软件中的“Logic Analyzer”分析工具进行仿真，可以更直观地观察端口的变化情况。执行“View”→“Symbol Window”命令，可以查看虚拟寄存器的情况。可以看到，LED 连接的 P2 端口的虚拟寄存器名称为 PORT2，如图 1-3-24 所示。

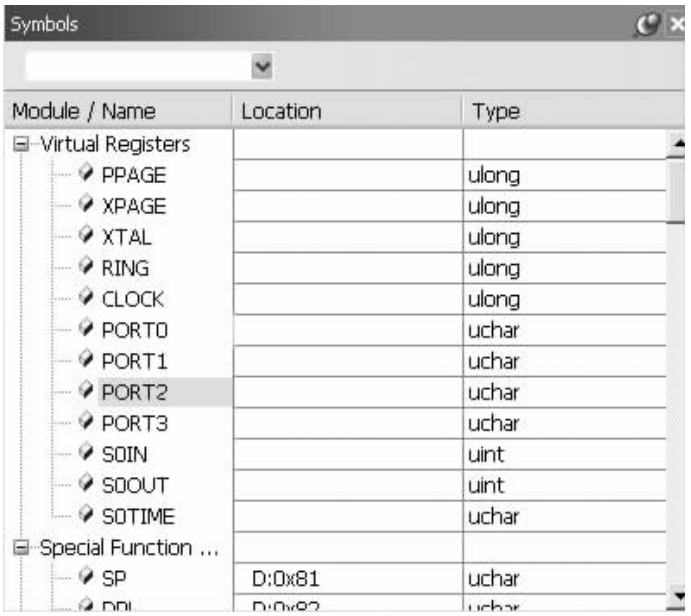


图 1-3-24　通过“Symbol Window”选项查看 P2 端口的虚拟寄存器

如图 1-3-25 所示，执行“View”→“Analysis Windows”→“Logic Analyzer”命令，打开虚拟逻辑分析仪。通过虚拟逻辑分析仪可以看到观察窗口，如图 1-3-26 所示。

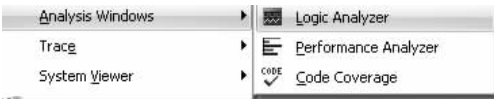


图 1-3-25　打开虚拟逻辑分析仪

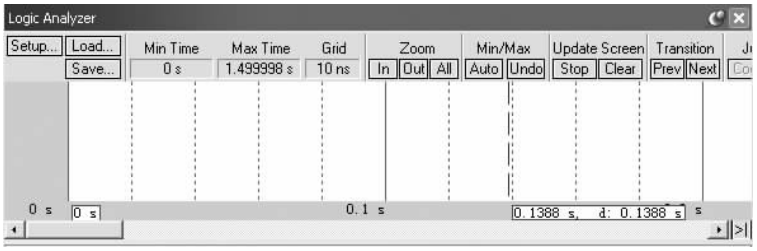


图 1-3-26　虚拟逻辑分析仪的观察窗口

单击“Setup”按钮，设置观察的引脚，如图 1-3-27 所示。单击右上角的按钮，可以对引脚进行添加。首先输入“PORT2.0”（不区分大小写）。

在“Display Type”（信号类型）下拉菜单中选择信号类型为“Bit”，如图 1-3-28 所示。这样就成功添加了引脚 P2.0。按照同样的方法，可以添加引脚 P2.1、P2.2。

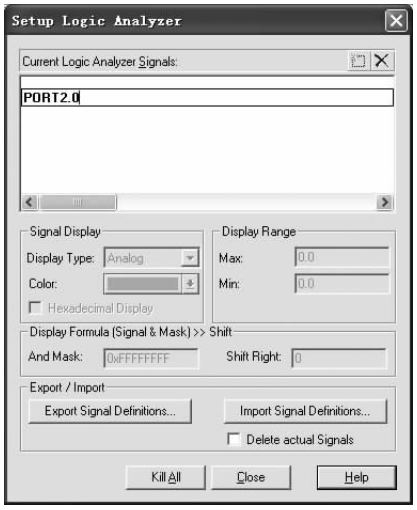


图 1-3-27　设置观察的引脚

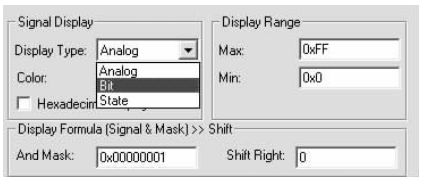


图 1-3-28　选择信号类型

引脚添加完毕，执行“Debug”→“Run”命令，在“Logic Analyzer”窗口可以看到信号的高低电平变化。为了更方便地观察信号，可以在程序运行一段时间后单击“Stop”按钮，停止运行程序。通过“Zoom”选项可以选择需要观察信号的区域，如图 1-3-29 所示。

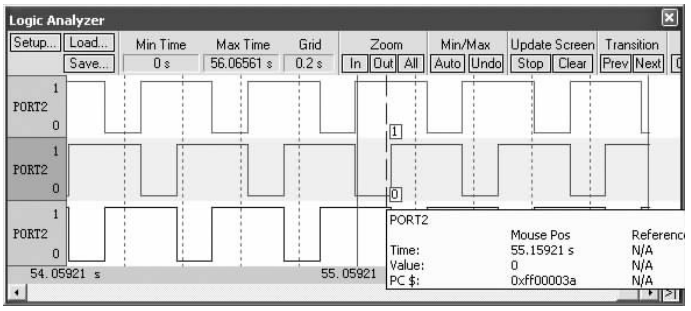


图 1-3-29　选择需要观察信号的区域

通过观察图 1-3-29，可以看到，引脚 P2.0、P2.1、P2.2 依次出现低电平，这个输出结果符合程序设计的要求。

1.4 Proteus 仿真环境

在程序通过编译与调试以后，可以利用 Proteus 仿真软件，对程序功能进行仿真调试。

1. 创建新工程

启动 Proteus 仿真软件，进入主界面，如图 1-4-1 所示。

图 1-4-1　Proteus 软件主界面

执行“File”→“New Project”命令，或者单击主界面中“Start”功能区的“New Project”按钮，创建新仿真工程。此时弹出对话框，设置工程名与保存路径，如图 1-4-2 所示。本例中，新仿真工程命名为 LED.pdsprj，保存路径与原 Keil 工程保存路径相同，即 E:/C51/LED。用户可根据自己的情况进行调整。

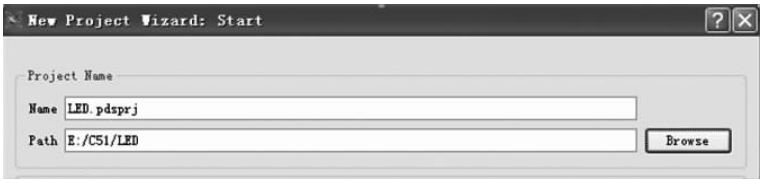


图 1-4-2　在 Proteus 软件中创建新工程

视频
Proteus 软件的使用

设置完成后，单击“Next”按钮，进入新工程原理图设计向导界面。选择“Create a schematic from the selected template”单选按钮，即从选定的模板中创建一个原理图，如图 1-4-3 所示。

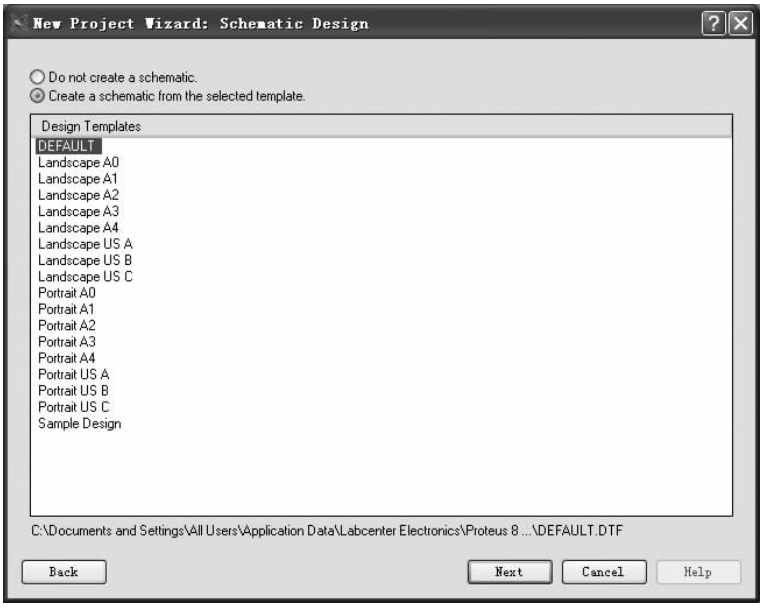


图 1-4-3　在新工程原理图设计向导界面中选择创建一个原理图

单击“Next”按钮进行下一步，选择“Do not create a PCB layout”，即选择不创建 PCB 图。再单击“Next”按钮进行下一步，选择“No Firmware Project”。再单击“Next”按钮进行下一步，单击“Finish”按钮，原理图仿真工程创建完毕，如图 1-4-4 所示。

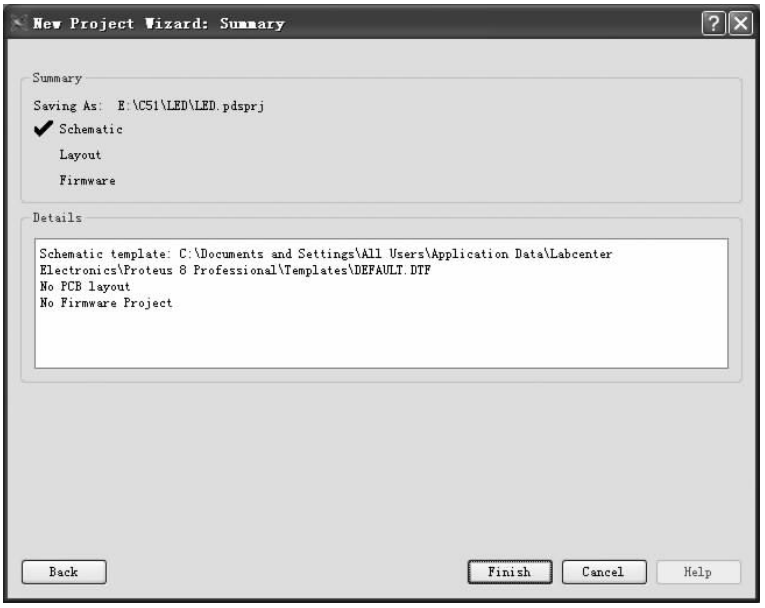


图 1-4-4　新工程创建完毕界面

2. 创建仿真原理图

启动 Proteus 8 Professional 软件，弹出原理图主界面，如图 1-4-5 所示。

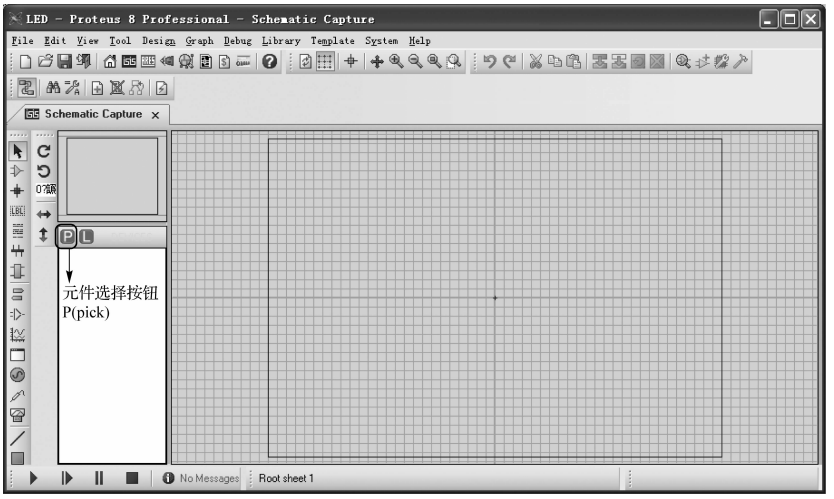


图 1-4-5　Proteus 8 Professional 原理图主界面

首先进行元件的选择，然后把元件添加到元件列表中。单击元件选择按钮“P(pick)”，弹出元件选择窗口，如图 1-4-6 所示。

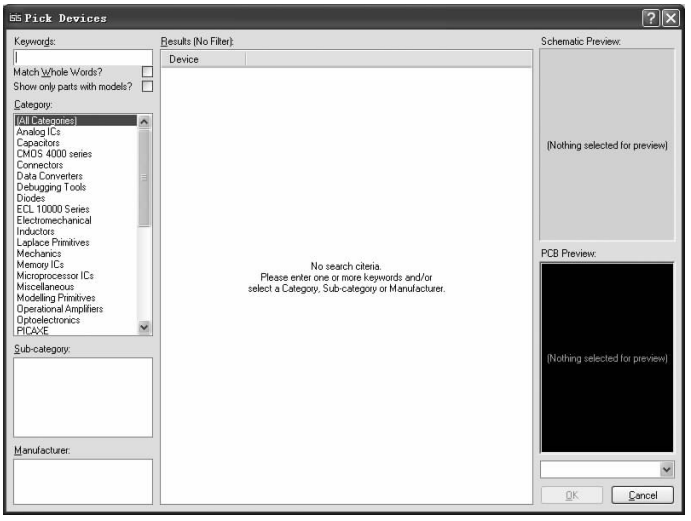


图 1-4-6　元件选择窗口

在左上角的“Keywords”文本框中输入需要的元件名称中的关键字，如图 1-4-7 所示。

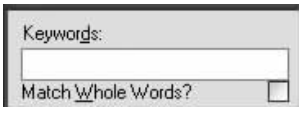


图 1-4-7　输入关键字选择元件

由于 Proteus 软件不支持 STC 系列的单片机仿真，因此选择兼容型号 AT89C52。此外，还需要电阻(Resistors)、黄色发光二极管(LED-YELLOW)。输入的名称是元件的英文名称，但不必输入完整的名称，输入相应关键字就能找到对应的元件。例如，在文本框中输

入“89C52”，得到以下结果，如图 1-4-8 所示。

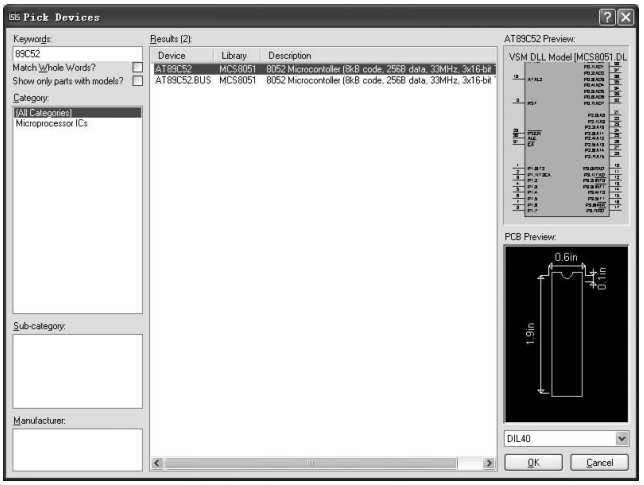


图 1-4-8　输入关键字“89C52”选择元件

在出现的搜索结果中双击需要的元件，该元件便会被添加到主窗口左侧的元件列表区。同理，输入“RES”，选择电阻；输入“LED”，选择“LED -YELLOW”。这样，AT89C52、电阻、黄色发光二极管都被添加到主窗口左侧的元件列表区。

3. 绘制电路图

在元件列表区单击选中 AT89C52，把光标移到右侧编辑窗口中，光标变成铅笔形状，单击，编辑窗口中出现一个 AT89C52 原理图的框图，可以移动。将光标移到合适的位置后，单击，原理图放置完毕。按照这个方法，依次将各个元件放置到绘图编辑窗口的合适位置，如图 1-4-9 所示。

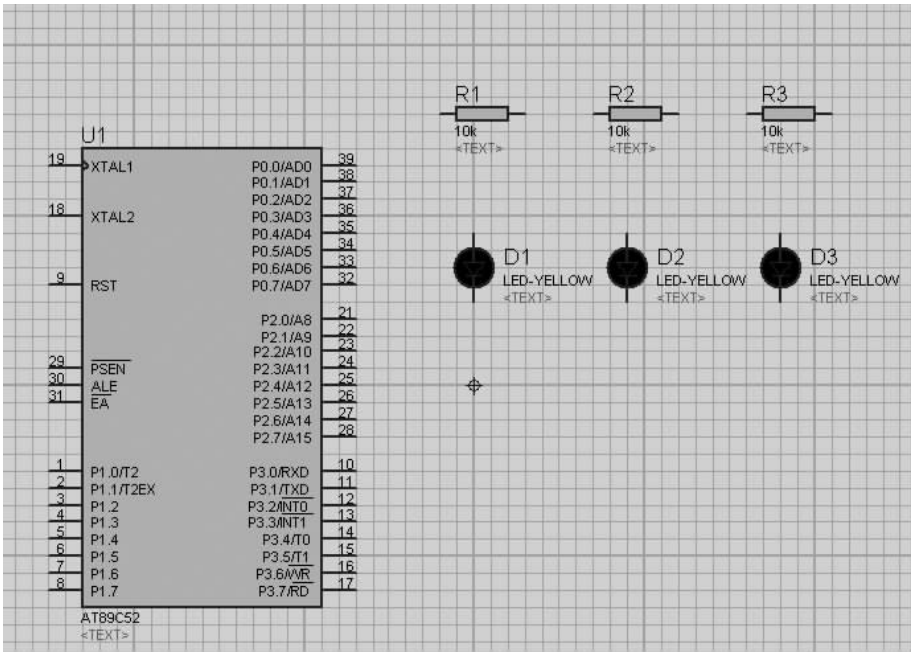


图 1-4-9　元件放置完成后的绘图编辑窗口

滚动鼠标滚轮便可对电路视图进行放大/缩小，视图会以光标为中心进行放大/缩小；绘图编辑窗口没有滚动条，只能通过预览窗口来调节绘图编辑窗口的可视范围。在预览窗口中移动绿色方框的位置即可改变绘图编辑窗口的可视范围。

为了方便原理图设计，需要调整电阻的方向。例如，在电阻 R1 处右击，如图 1-4-10 所示。

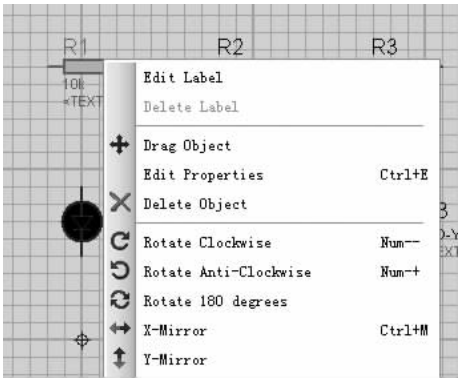


图 1-4-10　在电阻 R1 上右击所得快捷菜单

选择“Rotate Clockwise”命令，便可将 R1 顺时针旋转。同理，将 R2、R3 均旋转为垂直方向，如图 1-4-11 所示。

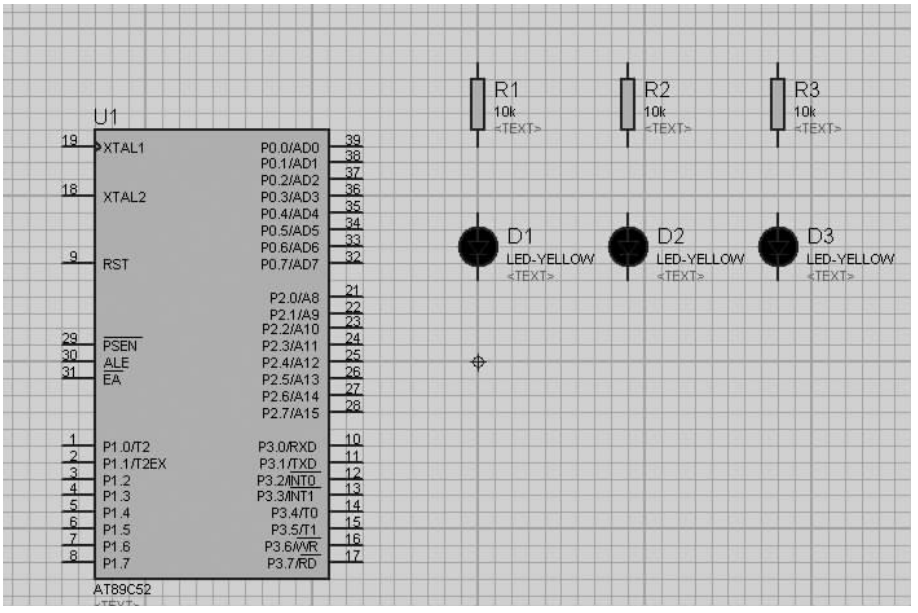


图 1-4-11　将电阻 R1、R2、R3 进行顺时针旋转后的绘图编辑窗口

接下来添加电源。单击模型选择工具栏中的图标，选择“POWER(电源)”，将其添加至绘图区，如图 1-4-12 所示。

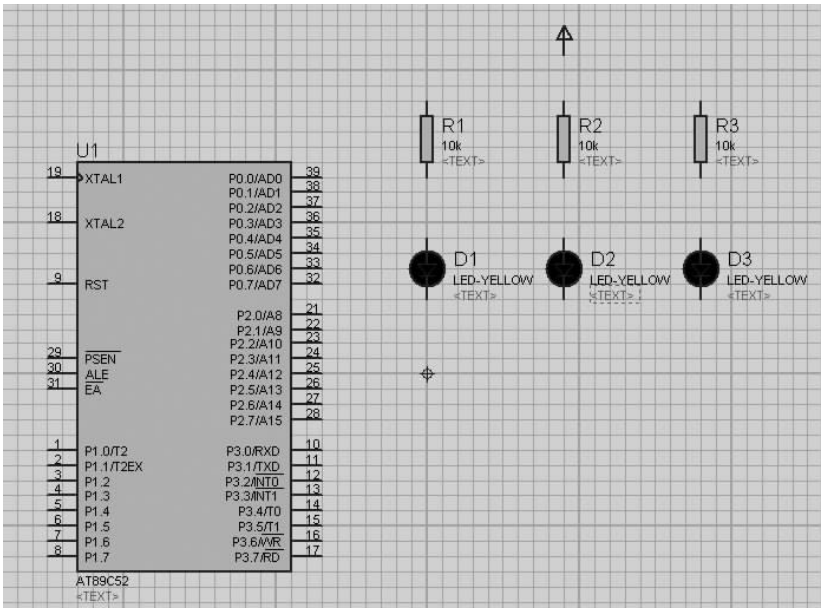


图 1-4-12　添加电源后的绘图编辑窗口

下面进行连线。将光标靠近元件的一端，当光标的铅笔形状变为绿色时，表示可以连线。单击该点，再将鼠标移至另一元件的一端，单击，两点间的线路就画好了。依次连接好所有线路（因为 Proteus 软件中单片机已默认提供电源，所以不用给单片机添加电源）。连线完毕的绘图编辑窗口如图 1-4-13 所示。

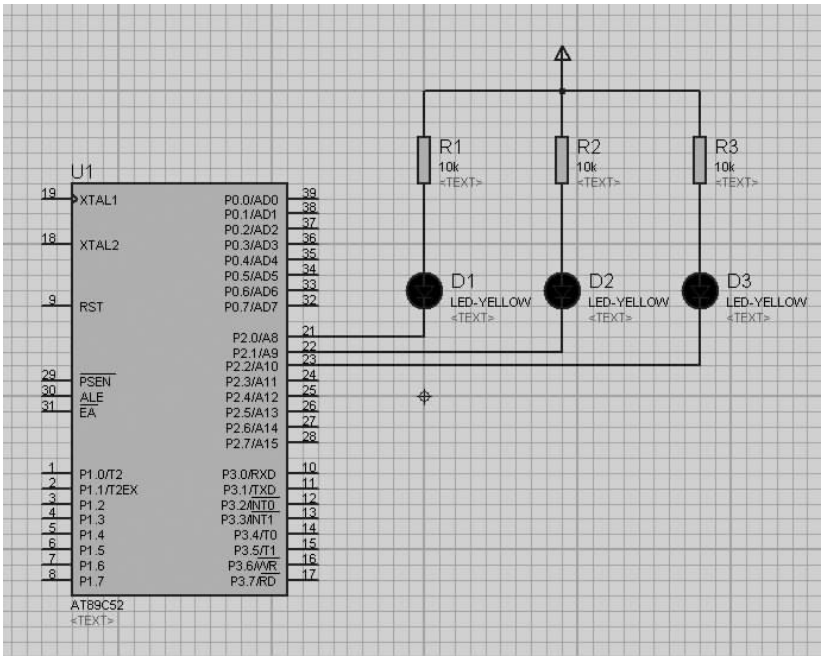


图 1-4-13　连线完毕的绘图编辑窗口

下面需要编辑元件，设置各元件参数。双击元件，弹出编辑元件的对话框，如图 1-4-14 所示。因为发光二极管点亮电流大小为 10 mA 左右，所以限流电阻设为 200 Ω。

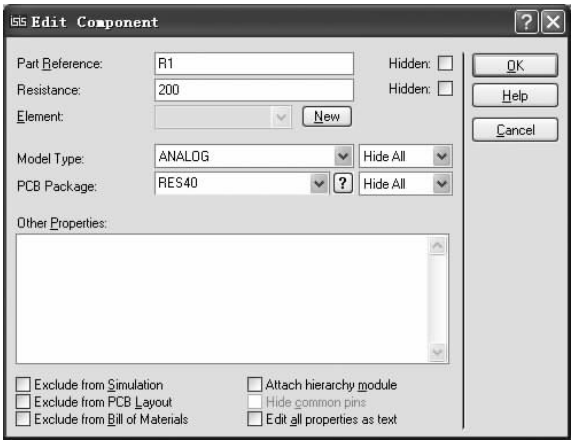


图 1-4-14　电阻参数设置对话框

双击单片机 AT89C52，弹出单片机参数设置对话框，如图 1-4-15 所示。

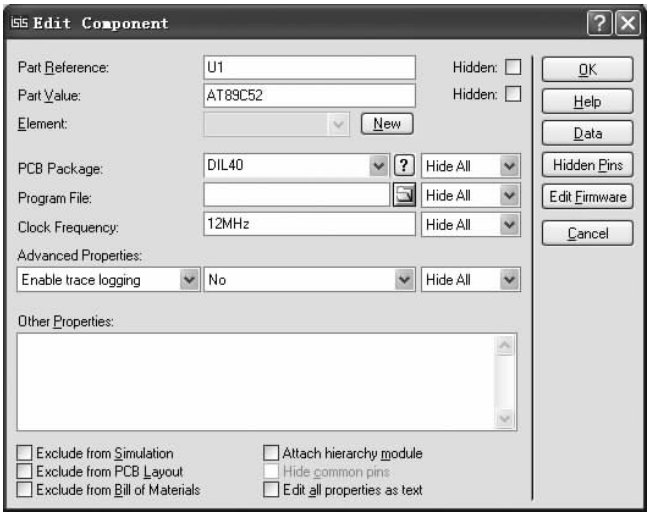


图 1-4-15　单片机参数设置对话框

单击“Program File”文本框后面的图标，选择要执行的程序，导入编好的程序“LED.hex”，如图 1-4-16 所示。

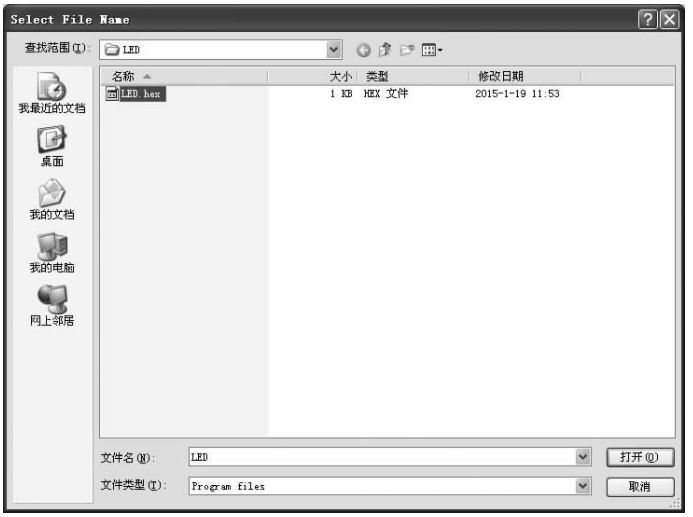


图 1-4-16　导入程序“LED. hex”

4. 仿真调试

在窗口左下方有仿真控制按钮。▶表示运行，⏯表示单步运行，⏸表示暂停，■表示停止。单击▶按钮，程序开始执行，LED 依次点亮。在运行时，电路中输出的高电平用红色表示，低电平用蓝色表示。绘图编辑窗口如图 1-4-17 所示。

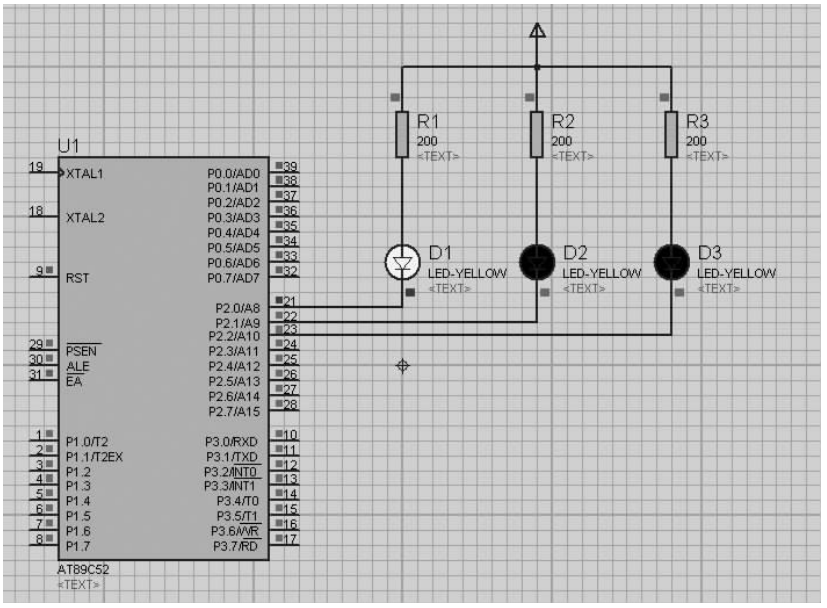


图 1-4-17　仿真调试中的绘图编辑窗口

在仿真过程中，可以看到 LED 交替闪烁的频率非常慢，不是设计要求的 500 ms。产生这一问题的原因是，在开发板中使用的单片机是单时钟/机器周期(1T)的 STC12C5A60S2，指令代码完全兼容传统 8051，但速度快 12 倍。而在 Proteus 仿真时，Proteus 器件库中没有 STC12C5A60S2，只能采用传统单片机 AT89C52 代替，因此在延时程序上消耗时间相差比较大。