项目一
单片机识别与展望

知识目标

了解各种单片机外部结构及封装形式；

了解 I/O 端口内部结构；

掌握各引脚的功能。

技能目标

掌握识别单片机各引脚的方法；

掌握各引脚的使用方法。

项目描述

本项目只限于单片机芯片的识别，芯片的制造和封装是微电子制造技术专业专门学习的内容。对于单片机应用的技术人员而言，只限于单片机如何应用，而且在应用时主要掌握芯片的使用方法、封装形式和各引脚的功能，封装形式在制作单片机产品的印制电路板时要用到，各引脚的功能在设计产品时用到。

相关知识

一、芯片引脚排列识别

单片机芯片封装有针脚式封装(DIP)与表面贴片式封装(SMD)两大类。制作单片机产品时,先按芯片封装形式在电子CAD软件中设计好印制电路板(PCB),然后送到专门制作PCB的厂家制作出PCB,再在焊接场地焊接好芯片,组装成电子产品。针脚式封装的元件体积较大,电路板必须钻孔才能安装元件,焊接时需要人工插入单片机芯片,再经过锡炉或喷锡(也可手焊)焊接好芯片。这种芯片成本较高,一般用于小批量的手工焊接产品。较新的设计都是采用体积小的表面贴片式元件,这种元件不必钻孔,用钢膜将半熔状锡膏倒在PCB上,再把芯片放上,用热风枪加热,即可将芯片焊接在PCB上。这种封装加工简单,焊接容易,成本较低,一般用于大批量的自动贴片机焊接产品。

单片机采用40引脚的双列直插封装方式时,封装及外形如图1-1所示,图1-1(a)为实物图,图1-1(b)为原理图库中的器件图,图1-1(c)为PCB封装库中的封装图。为了识别引脚,芯片都开有缺口,识别时将芯片缺口朝上,引脚向下放置于桌面上,左边第一脚即为引脚"1",有的芯片在开口的同时还在引脚"1"处刻有圆圈作为标记,按此标记逆时针依次排列引脚。

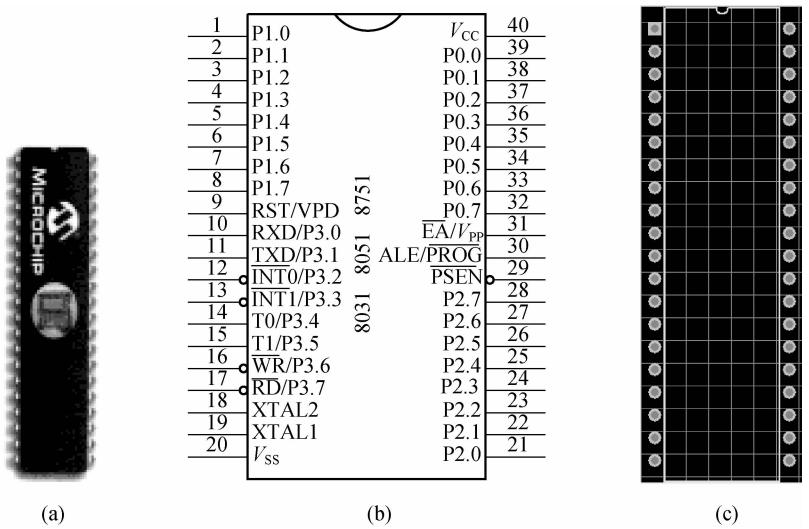


图1-1 单片机双列直插封装图

单片机贴片封装如图1-2所示,图1-2(a)为原理图库中的器件图,图1-2(b)为PCB封装库中的封装图。为了识别引脚,方形芯片有一个角被切除,识别时将芯片缺角朝左上方,引脚向下放置于桌面上,左边第一脚即为引脚"1",有的芯片在切角的同时还在引脚"1"处刻有圆圈作为标记。

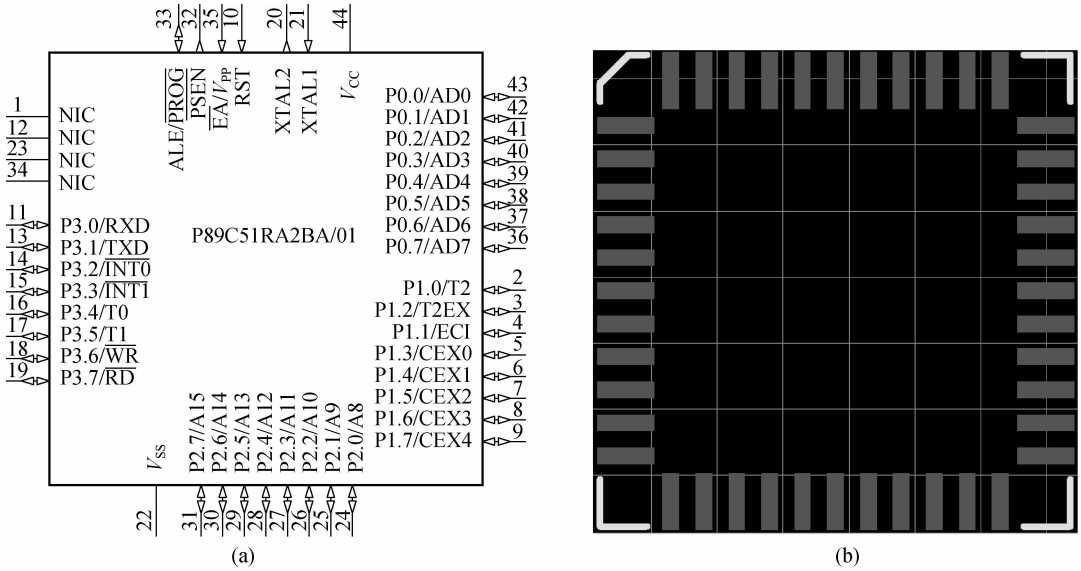


图 1-2　单片机贴片封装图

二、芯片引脚功能

以 51 芯片为内核的单片机现在派生出很多厂家，图 1-1 所示单片机的 40 个引脚功能说明如下。

1. 主电源引脚 V_{SS} 和 V_{CC}

1）V_{SS}

V_{SS}（第 20 引脚）接地。

2）V_{CC}

正常操作时，V_{CC}（第 40 引脚）为＋5 V 电源。

2. 外接晶振引脚 XTAL1 和 XTAL2

1）XTAL1

XTAL1（第 19 引脚）为内部振荡电路反相放大器的输入端，是外接晶体的一个引脚。当采用外部振荡器时，此引脚接地。

2）XTAL2

XTAL2（第 18 引脚）为内部振荡电路反相放大器的输出端，是外接晶体的另一个引脚。当采用外部振荡器时，此引脚接外部振荡源。

3. 控制或与其他电源复用引脚

1）RST/VPD

当振荡器运行时，在 RST/VPD（第 9 引脚）上出现两个机器周期的高电平（由低到高跳变），将单片机复位；在 V_{CC} 掉电期间，此引脚可接备用电源，由 VPD 向内部提供备用电源，以保持单片机内部 RAM 中的数据。

2）ALE/$\overline{PROG}$

正常操作时，ALE/$\overline{PROG}$（第30引脚）为ALE功能（允许地址锁存），用于把地址的低字节锁存到外部锁存器。ALE引脚以不变的频率（振荡器频率的1/6）周期性地发出正脉冲信号，可用做对外输出的时钟或用于定时。每当访问外部数据存储器时，将跳过一个ALE脉冲，ALE端可以驱动（吸收或输出电流）8个LSTTL负载。对于EPROM型单片机，在EPROM编程期间，此引脚接收编程脉冲（$\overline{PROG}$功能）。

3）$\overline{PSEN}$

$\overline{PSEN}$（第29引脚）为外部程序存储器读选通信号输出端，在从外部程序存储器读取指令（或数据）期间，$\overline{PSEN}$在每个机器周期内两次有效。$\overline{PSEN}$同样可以驱动8个LSTTL负载。

4）$\overline{EA}/V_{PP}$

$\overline{EA}/V_{PP}$（第31引脚）为内部程序存储器和外部程序存储器选择端。当$\overline{EA}/V_{PP}$为高电平时，访问内部程序存储器；当$\overline{EA}/V_{PP}$为低电平时，则访问外部程序存储器。对于EPROM型单片机，在EPROM编程期间，此引脚上加21 V编程电源（V_{PP}）。

4. 输入/输出引脚

1）P0口

P0口（P0.0～P0.7，对应芯片第39～32引脚）是一个8位漏极开路型双向I/O端口，在访问外部存储器时，它是分时传送的低字节地址和数据总线，P0口能以吸收电流的方式驱动8个LSTTL负载。

2）P1口

P1口（P1.0～P1.7，对应芯片第1～8引脚）是一个带有内部提升电阻的8位准双向I/O端口，能驱动（吸收或输出电流）4个LSTTL负载。

3）P2口

P2口（P2.0～P2.7，对应芯片第21～28引脚）是一个带有内部提升电阻的8位准双向I/O端口，在访问外部存储器时，它输出高8位地址。P2口可以驱动（吸收或输出电流）4个LSTTL负载。

4）P3口

P3口（P3.0～P3.7，对应芯片第10～17引脚）是一个带有内部提升电阻的8位准双向I/O端口，能驱动（吸收或输出电流）4个LSTTL负载。P3口还用于第二功能，具体内容参见表1-1。

图1-2所示贴片封装单片机共有44只引脚，空引脚有4只，其他引脚功能与上述相同，可自行学习。

三、I/O端口内部结构

I/O端口又称为I/O接口，也称为I/O通道或I/O通路，是MCS-51系列单片机对外部实现控制和信息交换的必经之路。I/O端口有串行和并行之分，串行I/O端口一次只能传送一位二进制信息，并行I/O端口一次能传送一组二进制信息。

1. 串行I/O端口

8051单片机有一个全双工的可编程串行I/O端口。这个串行I/O端口既可以在程序

控制下将 CPU 的 8 位并行数据变成串行数据一位一位地从发送数据线 TXD 发送出去，也可以把串行接收到的数据变成 8 位并行数据送给 CPU，而且这种串行发送和串行接收既可以单独进行，也可以同时进行。

8051 单片机串行发送和串行接收利用了 P3 口的第二功能，即利用 P3.1 引脚作为串行数据的发送线 TXD，P3.0 引脚作为串行数据的接收线 RXD，具体内容参见表 1-1。串行 I/O 端口的电路结构还包括串行口控制器 SCON、电源及波特率选择寄存器 PCON 和串行数据缓冲器 SBUF 等，它们都属于特殊功能寄存器。其中 PCON 和 SCON 用于设置串行口工作方式和确定数据的发送和接收波特率，SBUF 实际上由两个 8 位寄存器组成，一个用于存放待发送的数据，另一个用于存放接收到的数据，起着数据的缓冲作用，这些将在以后的项目中加以详细介绍。

2. 并行 I/O 端口

MCS-51 系列单片机设有 4 个 8 位双向 I/O 端口（P0 口、P1 口、P2 口、P3 口），每一个 I/O 端口都能独立地用做输入或输出。P0 口为三态双向口，能驱动 8 个 LSTTL 负载；P1 口、P2 口、P3 口为准双向口（在用做输入线时，端口锁存器必须先写入“1”，故称为准双向口），负载能力为 4 个 LSTTL 电路。

1）P0 口功能

P0 口位结构如图 1-3 所示，包括一个输出锁存器，两个三态缓冲器，一个输出驱动电路和一个输出控制端。输出驱动电路由一对场效应管组成，其工作状态受输出端的控制，输出控制端由一个与门、一个反相器和一个转换开关 MUX 组成。对 8051 单片机来讲，P0 口既可作为 I/O 端口，又可作为地址/数据总线使用。

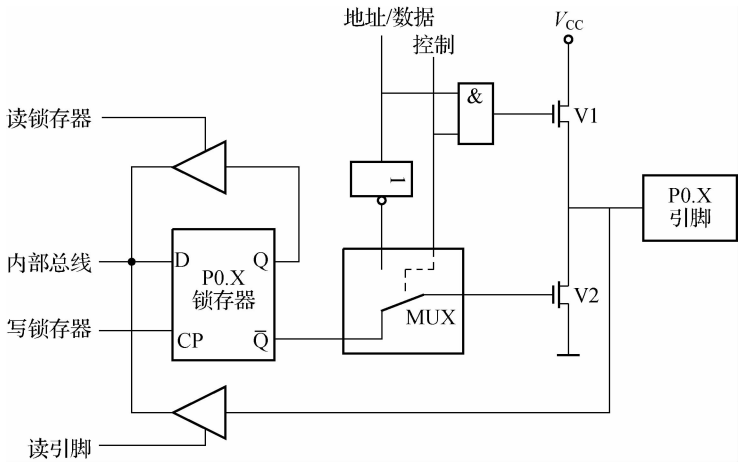


图 1-3　P0 口位结构

（1）P0 口作地址/数据总线使用。若从 P0 口输出地址或数据信息，此时控制端应为高电平，转换开关 MUX 将反相器输出端与输出级场效应管 V2 接通，同时与门开锁，内部总线上的地址或数据信号通过与门去驱动 V1 管，又通过反相器去驱动 V2 管，这时内部总线上的地址或数据信号就传送到 P0 口的引脚上。工作时低 8 位地址与数据线分时使用 P0 口。低 8 位地址由 ALE 信号的负跳变使它锁存到外部地址锁存器中，而高 8 位地址由 P2 口输出。

(2)P0口作通用I/O端口使用。对于有内部ROM的单片机,P0口也可以作通用I/O端口,此时控制端为低电平,转换开关把输出级与锁存器的Q端接通,同时因与门输出为低电平,V1管处于截止状态,输出级为漏极开路电路,在驱动NMOS电路时应外接上拉电阻。P0口作输入口用时,应先将锁存器写"1",这时输出级两个场效应管均截止,可作高阻抗输入,通过三态输入缓冲器读取引脚信号,从而完成输入操作。

(3)P0口上的"读—修改—写"功能。图1-3上面一个三态缓冲器是为了读取锁存器Q端的数据,Q端与引脚的数据是一致的,结构上这样安排是为了满足"读—修改—写"指令的需要。这类指令的特点是:先读锁存器,随之可能对读入的数据进行修改再写入端口。这类指令同样适合P1口、P2口和P3口,其操作是:先将端口的全部8位数读入,再通过指令修改某些位,然后将新的数据写回到锁存器中。

2)P1口功能

P1口是一个有内部上拉电阻的准双向口,位结构如图1-4所示。

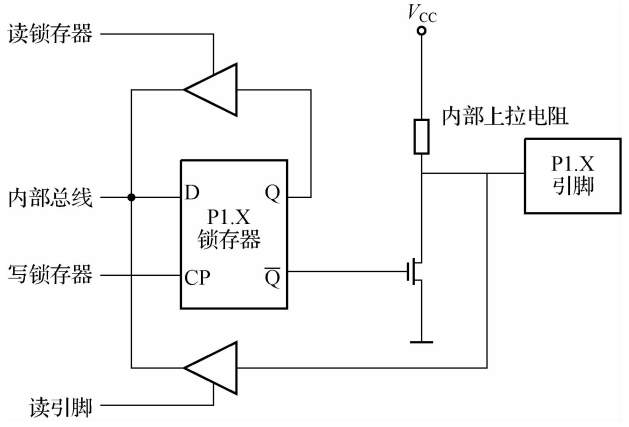


图1-4 P1口位结构

(1)P1口作通用I/O端口使用。P1口的每一只引脚都能独立用做输入线或输出线。作输出时,将"0"写入锁存器,场效应管导通,输出线为低电平,即输出为"0"。作输入时,必须先将"1"写入锁存器,使场效应管截止。该引脚由内部上拉电阻提拉成高电平,同时也能被外部输入源拉成低电平,即当外部输入"1"时该引脚为高电平,而输入"0"时,该引脚为低电平。P1口作输入时,可被任何TTL电路和MOS电路驱动,由于具有内部上拉电阻,也可以直接被集电极开路和漏极开路电路驱动,不必外加上拉电阻。

(2)P1口其他功能。P1口在EPROM编程和验证程序时,它输入低8位地址。在8032/8052单片机中,P1.0引脚和P1.1引脚是多功能的:P1.0引脚可作定时/计数器2的外部计数触发输入端,P1.1引脚可作定时/计数器2的外部控制输入端。

3)P2口功能

P2口位结构如图1-5所示,引脚上拉电阻同P1口。在结构上,P2口比P1口多一个输出控制部分。

(1)P2口作通用I/O端口使用。当P2口作通用I/O端口使用时,是一个准双向口,此时转换开关MUX倒向左边,输出级与锁存器接通,引脚可接I/O设备,其I/O操作与P1口完全相同。

(2)P2 口作地址总线使用。当系统中接有外部存储器时,P2 口用于输出高 8 位地址 A15～A8。这时在 CPU 的控制下,转换开关 MUX 倒向右边,接通内部地址总线。P2 口的引脚状态取决于片内输出的地址信息,这些地址信息来源于 PCH、DPH 等。在外接程序存储器的系统中,由于访问外部存储器的操作连续不断,P2 口不断送出地址高 8 位。在 8031 单片机构成的系统中,P2 口一般只作地址总线使用,不作 I/O 端口直接连外部设备。

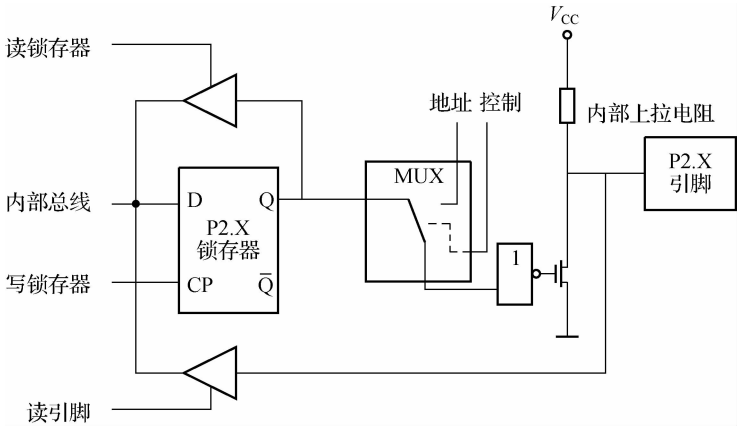


图 1-5　P2 口位结构

4)P3 口功能

P3 口是一个多用途的端口,也是一个准双向口,作为第一功能使用时,其功能同 P1 口。P3 口位结构如图 1-6 所示。

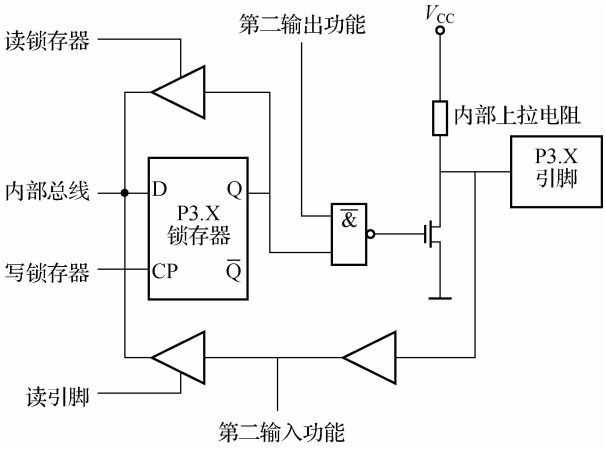


图 1-6　P3 口位结构

作为第二功能使用时,P3 口每一位功能定义见表 1-1。P3 口的第二功能实际上就是系统具有控制功能的控制线,此时相应 P3 口的锁存器必须为“1”状态。与非门的输出由第二功能输出线的状态确定,从而 P3 口的状态取决于第二功能输出线的电平。在 P3 口的引脚信号输入通道中有两个三态缓冲器,第二功能的输入信号取自第一个缓冲器的输出端,第二个缓冲器仍是第一功能的读引脚信号缓冲器。

表 1-1　P3 口的第二功能

引脚功能	第二功能
P3.0	RXD——串行输入(数据接收)口
P3.1	TXD——串行输出(数据发送)口
P3.2	$\overline{\text{INT0}}$——外部中断 0 输入
P3.3	$\overline{\text{INT1}}$——外部中断 1 输入
P3.4	T0——定时器 0 外部输入
P3.5	T1——定时器 1 外部输入
P3.6	$\overline{\text{WR}}$——外部数据存储器写选通信号输出
P3.7	$\overline{\text{RD}}$——外部数据存储器读选通信号输入

每个 I/O 端口内部都有一个 8 位数据输出锁存器和一个 8 位数据输入缓冲器，四个数据输出锁存器与端口号 P0、P1、P2 和 P3 同名，皆为特殊功能寄存器。因此，CPU 数据从并行 I/O 端口输出时可以得到锁存，数据输入时可以得到缓冲。

四个并行 I/O 端口作为通用 I/O 端口使用时，共有写端口、读端口和读引脚三种操作方式。写端口实际上就是输出数据，是将累加器 A 或其他寄存器中的数据传送到端口锁存器中，然后自动从端口引脚线上输出。读端口不是真正的从外部输入数据，而是将端口锁存器中的输出数据读取后存到 CPU 的累加器。读引脚才是真正的输入外部数据的操作，是从端口引脚线上读取外部的输入数据。I/O 端口的上述三种操作是通过指令或程序来实现的，这些将在以后的项目中详细介绍。

项目实践

1. 参观元器件买卖市场

熟悉当地元器件买卖市场，自己购买各种集成块，查看并了解各种元器件。

2. 熟悉各种元器件网站，会查各种资料

搜索三家以上元器件销售网站，了解多家集成块生产厂家，最低收集 10 家，了解各生产厂家的产品质量和价格，以及每种器件的技术参数、制作工艺、制造设备等知识。下载各种资料，增加自己相关知识。与此同时，在计算机中安装各种仿真软件、课件和动画，以供学习钻研，多做多练，培养学习能力和动手能力。

3. 熟悉元器件

认真查看购买回来的单片机集成块，比较不同集成块之间的主要区别。用万用表测试每一个脚对地(GND)的电阻，记录下每个阻值，便于以后检修，因为集成块损坏时这些阻值会发生改变。

拓展知识

单片微型计算机简称为单片机。它是微型计算机发展中的一个重要分支，它以其独特

的结构和性能，越来越广泛地应用到工业、农业、国防、网络、通信以及人们的日常工作、生活领域中。

单片机在一块芯片上集成了中央处理部件(CPU)、存储器(RAM、ROM)、定时/计数器和各种I/O端口(并行I/O端口、串行I/O端口和A/D转换器)等部件。由于单片机通常是为实时控制应用而设计和制造的，因此，又称为微控制器(MCU)。

1. 单片机的基本知识

每一种单片机的设计都包括以下几个方面。

(1)指令及与指令对应的电路(芯片)和代码。

(2)固化程序的编程器。

(3)硬件仿真器。

设计完成后，由有关生产厂家生产出产品(芯片、编程器、仿真器)，再由开发人员开发出市场所需要的应用产品。在这些开发的芯片中，有些芯片是公开使用的，在市场上能买到的芯片就属这种类型；有些芯片是不公开的，如军工产品和各大公司开发的专用产品。公开使用的芯片又分为两类：一类不能加密，一类可加密。

单片机种类很多，但不管哪种单片机，厂家都要配套提供编程器(固化程序用)、硬件仿真器(调试程序用)、指令系统和芯片使用说明书，没有这些很难进行二次开发，除非能破解芯片。因而，对产品开发人员来说，所要做的工作就是按厂家提供的方法使用芯片，并且能够按产品功能要求设计电路、编写程序、做成产品。对产品维修使用人员来说，知道芯片的使用方法和产品电路的工作原理，会维修使用即可。

2. 单片机的发展概况

单片机自问世以来，性能不断提高和完善，其资源不仅能满足很多应用场合的需要，而且具有集成度高、功能强、速度快、体积小、功耗低、使用方便、性能可靠、价格低廉等特点，因此，在工业控制、智能仪器仪表、数据采集和处理、通信系统、网络系统、汽车工业、国防工业、高级计算器具、家用电器等领域的应用日益广泛，并且正在逐步取代现有的多片微机应用系统。单片机的潜力越来越被人们所重视，特别是当前用CMOS工艺制成的各种单片机，由于功耗低、使用的温度范围大、抗干扰能力强、能满足一些特殊要求的应用场合等特点，不仅扩大了单片机的应用范围，也进一步促进了单片机技术的发展。

自1976年9月Intel公司推出MCS-48系列单片机以来，单片机就受到了广大用户的欢迎。因此，有关公司都争相推出各自的单片机，如GI公司推出的PIC1650系列单片机，Rockwell公司推出的与6502微处理器兼容的R6500系列单片机，它们都是8位机，片内有8位中央处理器(CPU)、并行I/O端口、8位定时/计数器和容量有限的存储器(RAM、ROM)以及简单的中断功能。

1980年，Intel公司在MCS-48系列单片机的基础上又推出了高性能的MCS-51系列单片机。这类单片机均带有串行I/O端口，定时/计数器为16位，片内存储容量(RAM、ROM)相应增大，并有优先级中断处理功能。MCS-51系列单片机扩大了功能和寻址范围，是当时单片机应用的主流产品。

1982年，Mostek公司和Intel公司先后又推出了性能更高的16位单片机MK68200和MCS-96系列，NS公司和NEC公司也分别在原有8位单片机的基础上推出了16位单片机

HPC16040和μPD783××系列。1987年Intel公司又推出了性能比8096单片机高两倍的CMOS型单片机80C196,1988年推出带EPROM的87C196单片机。由于16位单片机的推出时间较迟、价格昂贵、开发设备有限等多种原因,至今还未得到广泛应用。而8位单片机已能满足大部分应用的需要,因此,在推出16位单片机的同时,高性能的新型8位单片机也不断问世。

目前国际市场上8位、16位单片机系列已有很多,但是,在国内使用较多的系列是Intel公司的产品,其中又以MCS-51系列单片机应用尤为广泛,三十几年经久不衰,而且还在进一步发展完善,价格越来越低,性能越来越好。单片机技术正以惊人的速度向前发展,就市场上已出现的单片机而言,其技术革新与进步主要表现在以下几个方面。

1)CPU的发展

增加CPU的字长或提高时钟频率均可提高CPU的数据处理能力和运算速度。CPU的字长目前有8位、16位和32位,时钟频率高达20 MHz的单片机也已出现,还有的8位单片机其算术逻辑运算部件(ALU)是16位,内部采用16位数据总线。另外,单片机内部采用双CPU结构能大大提高处理能力,由于片内有两个CPU能同时工作,能更好地处理外围设备的中断请求,克服了单CPU在多重高速中断响应时的失效问题。同时,由于双CPU可以共享存储器和I/O端口的资源,因此,还可更好地解决信息通信问题,如Intel公司的8044单片机,它的内部实际上是8051单片机和SIU通信处理机组成,由SIU来管理SDLC的通信,这样既加快了通信处理的速度,同时,还减轻了8051单片机的处理负担。

2)片内存储器的发展

(1)扩大存储容量。早期单片机的片内存储器,一般RAM为64～128 B,ROM为1～2 KB,寻址范围为4 KB。新型单片机片内RAM为256 B,ROM多达16 KB。新型单片机的寻址范围可扩大到64 KB,甚至128 KB(其中随机存储器RAM容量为64 KB,只读存储器ROM容量为64 KB)。内部ROM分可擦除和一次性可编程(OTP)两种,前者价高,技术开发时使用,后者价低,开发成功后,一次性固化在产品上使用,须注意的是一次性固化在产品上使用的必须是成熟产品,否则会造成经济损失。

(2)片内EPROM开始E^2PROM化。早期单片机内ROM有的采用可擦除式的只读存储器——EPROM,然而EPROM必须要高压编程,紫外线擦除,给使用带来不便。近年来推出的电擦除可编程只读存储器——E^2PROM,可在正常工作电压下进行读写,并能在断电的情况下保持信息不丢失,因此,有些厂家已开始用E^2PROM替代原来的片内EPROM。由于写入E^2PROM的数据能永久保存,因此,有些厂家已开始将E^2PROM用做片内ROM,甚至用做片内通用寄存器,这样可省去备用电池。

(3)闪速存储器。随着CMOS工艺的改进和提高,闪速存储器在不断发展和完善,应用越来越广,容量越来越大,价格越来越低。闪存技术在各个领域得到应用,如ATMEL公司将闪存技术应用到单片机中,生产出了带闪速存储器的AT89系列单片机,对一些小系统,外部可以不用扩展存储芯片,从而使得只用单片机就能构成一个完整的控制系统。PIC系列单片机也有带闪存的存储芯片。

(4)串行存储器。I^2C总线的快速发展,使得串行数据存储器在容量和存储速度上有了很大的提高,由于它体积小,引脚少,价格低,因此得到了广泛的应用。

(5)片内程序的保密措施。为了使片内EPROM(E^2PROM)内容不被复制,一些厂家对

片内 EPROM(E^2PROM)采用加锁技术。如 Intel 公司的 8X252 单片机,加锁后的 EPROM(E^2PROM)中的程序只能供片内 CPU 读取,不能从片外读取,否则必须先开锁,开锁时,CPU 先自动擦除 EPROM(E^2PROM)中的信息,从而达到程序保密的目的。

3)片内 I/O 端口功能

最初的单片机,片内只有并行 I/O 端口、定时/计数器,它们的功能较弱,实际应用中往往需要通过特殊的接口扩展功能,从而也增加了应用系统结构的复杂性。

近年来,新型单片机内的接口,从类型和数量上都有很大的发展,这不仅大大提高了单片机的功能,而且使系统的总体结构也大大简化了。如有些单片机的并行 I/O 端口能直接输出大电流和高电压,可直接用于驱动荧光显示管(VFD)、液晶显示器(LCD)和数码显示管(LED)等,应用系统中就不再需要外部驱动电路。又如有些单片机片内含有 A/D 转换器,在一些实时控制系统中可省掉外部 A/D 转换器。

目前,在单片机中已出现的各类新型接口有数十种,如 A/D 转换器、D/A 转换器、DMA 控制器、CRT 控制器、LCD 驱动器、LED 驱动器、VFD 驱动器、正弦波发生器、声音发生器、字符发生器、波特率发生器、锁相环、频率合成器、脉宽调制器等。虽然一个单片机内只含若干种接口,但其功能却比初期的单片机强得多,因此,可用它作为高速主机的通用外设接口。

目前,单片机种类繁多,功能多样,将外围电路尽量集中在芯片内,使其成为名副其实的单片机,这也成为一种发展趋势。

4)单片机在工艺上的提高

单片机的制造工艺直接影响其性能。早期的单片机采用 PMOS 工艺,随后逐渐采用 NMOS、HMOST 和 CMOS 工艺。目前 8 位单片机中有一半产品已 CMOS 化,16 位单片机也已开始推出 CMOS 型产品。采用 CMOS 工艺的单片机,其工作电源范围较宽,功耗大小与电源电压成正比,所以降低电源电压即可降低功耗,但是降低电压会减慢指令执行速度,即降低单片机的运算速度。故一般希望在一定速度的前提下,尽量降低工作电压,减小功耗。

随着新型单片机片内接口电路的增多,外引脚也增多,为减少外引脚线,目前主要采用两种方式:一种是采用新颖的通信总线以减少外引线,一种是改进外封装,如采用扁平引脚封装 FP(flat pachage)、方形引脚封装 QIP(quad in line package)、叠背式封装 PBP(piggk back package)等。

5)片内固化应用软件和系统软件

将一些应用软件和系统软件固化于片内 ROM 中,可以简化应用程序的编制工作,为开发和应用提供方便。如 RUPI-44 系列单片机,把通信控制软件固化在片内,使通信程序大大简化。又如 Intel 公司在有的 MCS-51 系列单片机内固化了 PL/M-51 语言,在 8052 BH 单片机中固化了 BASIC 解释程序,不仅可用汇编语言,还可用 BASIC 语言编程,其 BASIC 语言系统比基本 BASIC 有所扩充,增加了很多适合控制用的语句、命令、运算符等,而且还允许 BASIC 语言和汇编语言互相调用,需要快速控制时,可用汇编语言,如采样、A/D 转换等,在作复杂的数据运算时,可用汇编语言来调用 BASIC 中现成的运算子程序,既能满足速度方面的要求,又能简化编程。

单片机的技术还在不断发展,新型单片机还将不断涌现,当前单片机的产量占整个微机(包括一般的微处理器)产量的 80% 以上。在我国,低档 8 位单片机(如 8048 单片机)于 20

世纪80年代初就开始应用，目前已转向高档8位单片机(MCS-51系列单片机、Z8系列单片机等)的应用，也有不少单位已转向16位单片机的开发和应用。

3.单片机的应用

单片机在一块芯片上集成了一台微型计算机所需的CPU、存储器、I/O部件和时钟电路等，因此它具有体积小、使用灵活、成本低、易于产品化、抗干扰能力强、可在各种恶劣环境下可靠地工作等特点，特别是它应用面广、控制能力强，使它在工业控制、智能仪表、外设控制、家用电器、机器人、军事装置等方面得到了广泛的应用。

单片机主要用于以下几方面。

1)控制系统中的应用

控制系统的工作环境恶劣，各种干扰也强，特别是工业控制系统，而且往往要求实时控制，故要求控制系统工作稳定、可靠、抗干扰能力强。单片机最适合用于控制领域，如炉子恒温控制、电镀生产线自动控制等。

2)智能仪表中的应用

用单片机制作测量、控制仪表，能使仪表向数字化、智能化、多功能化和柔性化发展，并使监测、处理、控制等功能一体化，使仪表重量大大减轻，便于携带和使用，同时降低了成本，提高了性价比，如数字式RLC测量仪、智能转速表、计时器等。

3)智能产品

单片机与传统的机械产品结合，使传统机械产品结构简化、控制智能化，构成新型的机、电、仪一体化产品，如数控车床、智能电动玩具、各种家用电器和通信设备等。

4)在智能计算机外设中的应用

在计算机应用系统中，除通用外部设备(键盘、显示器、打印机)外，还有许多用于外部通信、数据采集、多路分配管理、驱动控制等的接口。如果这些外部设备和接口全部由主机管理，势必造成主机负担过重、运行速度降低，并且不能提高对各种接口的管理水平。如果采用单片机专门对接口进行控制和管理，则主机和单片机就能并行工作，这不仅能大大提高系统的运算速度，而且单片机还可对接口信息进行预处理，以减少主机和接口间的通信密度，提高接口控制管理的水平，如绘图仪控制器，磁带机、打印机的控制器等。

目前，国外的单片机应用已相当普及，国内虽然从1980年开始才着手开发应用，但至今也已拥有数十家专门生产单片机开发系统的工厂或公司，越来越多的科技工作者投身到单片机的开发和应用中，并且在程序控制、智能仪表等方面涌现出大量的科技成果，可以预见，单片机在我国必将有更为广阔的发展前景。

复习思考题

1.什么是单片机？它与一般微型计算机在结构上有何区别？

2.单片机的发展大致可分为几个阶段？各阶段的单片机功能特点是什么？

项目二
单片机开发工具

知识目标

熟悉 Keil μVision3 软件各菜单的功能；

熟悉 Proteus 软件的基本操作；

了解 QTH 系列单片机仿真器开发应用程序的过程。

技能目标

掌握软件仿真开发软件的使用方法；

掌握硬件仿真开发软件的使用方法；

掌握在线硬件仿真开发系统的使用方法。

项目描述

单片机是一门综合性学科，需要模拟电子技术和数字电子技术作为先导知识，还要结合计算机使用基础知识，同时，单片机又是一门实践性很强的专业技术，使用的设备较多。对于初学者来说，没有进行理论学习、软件仿真、硬件仿真、程序固化、实物制作的全过程，很难入门，所以在学习过程中首先要掌握单片机开发工具。

相关知识

一、软件仿真

Keil C51 软件是 MCS-51 系列单片机应用开发软件，其模拟调试功能很强，基本上包括

了在线仿真器的单步、跟踪、检查和修改功能，并且还能模拟产生各种中断和I/O端口应答过程。Keil C51软件集可视化编辑、编译、调试、仿真于一体，支持汇编、PLM和C语言的混合编程，功能强大、界面友好、易学易用。下面主要介绍Keil μVision3软件的使用方法。

1. Keil μVision3 软件概述

Keil μVision3软件是流行的单片机开发环境之一，其安装的方法同一般的软件安装。安装完成后将在Windows桌面生成一个Keil μVision3的图标。单击“开始”→“程序”→Keil μVision3即可运行该软件，也可双击Keil μVision3的图标运行该软件。Keil μVision3的主窗口如图2-1所示，有菜单栏、工具栏、源代码窗口、项目窗口和输出窗口。

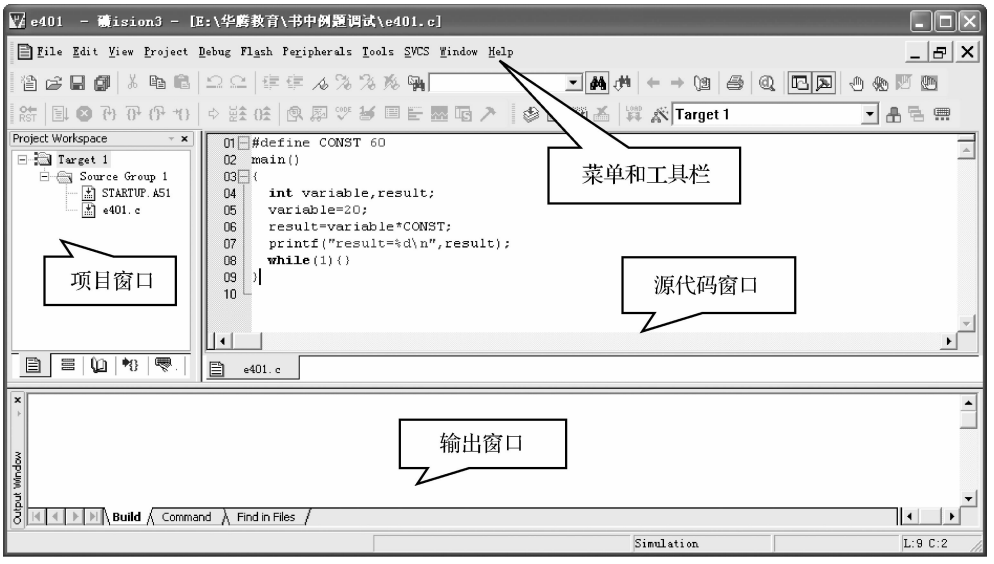


图2-1 Keil μVision3 主窗口

1）Keil μVision3软件的主菜单

Keil μVision3软件的主菜单提供各种操作菜单，如编辑操作、项目维护、开发工具选项设置、调试程序、窗口选择和处理以及在线帮助等，下面分别介绍。

（1）File菜单和命令见表2-1。

表2-1 File菜单和命令

菜　单	工具条	快捷键	描　述
New		Ctrl+N	创建新文件
Open		Ctrl+O	打开已经存在的文件
Close	—	—	关闭当前文件
Save		Ctrl+S	保存当前文件
Save all		—	保存所有文件
Save as	—	—	另外取名保存
Device Database	—	—	维护器件库
Print Setup	—	—	打印页面属性设置

续表

菜　单	工具条	快捷键	描　述
Print		Ctrl＋P	打印当前文件
Print Preview	—	—	打印预览
Exit	—	—	退出 Keil μVision3

(2)Edit 菜单和命令见表 2-2。

表 2-2　Edit 菜单和命令

菜　单	工具条	快捷键	描　述
Undo		Ctrl＋Z	取消上次操作
Redo		Ctrl＋Shift＋Z	重复上次操作
Cut		Ctrl＋X	剪切所选文本
Copy		Ctrl＋C	复制所选文本
Paste		Ctrl＋V	粘贴
Indent Selected Text		—	右移一个制表键的距离
Unindent Selected Text		—	左移一个制表键的距离
Toggle Bookmark		Ctrl＋F2	设置/取消当前行的标签
Goto Next Bookmark		F2	移动光标到下一个标签处
Goto Previous Bookmark		Shift＋F2	移动光标到上一个标签处
Clear All Bookmarks		—	清除当前文件的所有标签
Find		Ctrl＋F	在当前文件中查找文本
Replace	—	Ctrl＋H	替换特定的字符
Find in Files		Ctrl＋ Shift＋F	在多个文件中查找
Incremental Find		Ctrl＋I	增量式查找
Goto Matching Brace	—	—	转到对应的括号

(3)View 菜单和命令见表 2-3。

表 2-3　View 菜单和命令

菜　单	工具条	描　述
Status Bar	—	显示/隐藏状态
File Toolbar	—	显示/隐藏文件工具栏
Build Toolbar	—	显示/隐藏编译工具栏
Debug Toolbar	—	显示/隐藏调试工具栏
Project Window		显示/隐藏项目窗口

续表

菜　单	工具条	描　述
Output Window		显示/隐藏输出窗口
Source Browser		打开资源浏览器
Disassembly Window		显示/隐藏反汇编窗口
Watch & Call Stack Window		显示/隐藏观察和堆栈窗口
Memory Window		显示/隐藏存储器窗口
Code Coverage Window		显示/隐藏代码报告窗口
Performance Analyzer Window		显示/隐藏性能分析窗口
Symbol Window	—	显示/隐藏字符变量窗口
Serial Window #1		显示/隐藏串口1的观察窗口
Toolbox		显示/隐藏自定义工具
Periodic Window Update	—	程序运行时刷新调试窗口
Workbook Mode	—	显示/隐藏窗口框架模式
Include Dependencies	—	包含
Options	—	设置颜色、字体、快捷键等

(4)Project菜单和命令见表2-4。

表2-4　Project菜单和命令

菜　单	工具条	快捷键	描　述
New Project	—	—	创建新项目
Import μVision1 Project	—	—	转化μVision1的项目
Open Project	—	—	打开一个已经存在的项目
Close Project	—	—	关闭当前的项目
Target Environment	—	—	定义工具、包含文件和库的路径
Targets Groups Files	—	—	维护一个项目的文件组和文件
Select Device for Target	—	—	选择对象的CPU
Remove	—	—	从项目中移走一个组或文件
Options		Alt+F7	设置对象、组或文件的工具选项
File Extensions		—	选择不同文件类型的扩展名
Build Target		F7	编译修改过的文件并生成应用
Rebuild all Target Files		—	重新编译所有的文件并生成应用
Translate		Ctrl+F7	编译当前文件
Stop Build	—	—	停止生成应用的过程

(5)Debug 菜单和命令见表 2-5。

表 2-5　Debug 菜单和命令

菜　单	工具条	快捷键	描　述
Start/Stop Debugging		Ctrl+F5	开始/停止调试模式
Go		F5	运行程序,直到遇到一个中断
Step		F11	单步执行程序,进入子程序运行
Step over		F10	单步执行程序,跳过子程序
Step out of Current function	—	Ctrl+F11	执行到当前函数的结束
Stop Running		ESC	停止程序运行
Breakpoints	—	—	打开断点对话框
Insert/Remove Breakpoint		—	设置/取消当前行的断点
Enable/Disable Breakpoint		—	使能/禁止当前行的断点
Disable All Breakpoints		—	禁止所有的断点
Kill All Breakpoints		—	取消所有的断点
Show Next Statement		—	显示下一条指令
Enable/Disable Trace Recording		—	使能/禁止程序运行轨迹
View Trace Records	—	—	显示程序运行过的指令
Memory Map	—	—	打开存储器空间配置对话框
Performance Analyzer	—	—	打开设置性能分析的窗口
Inline Assembly	—	—	对某行重新汇编,修改汇编代码
Function Editor	—	—	编辑、调试函数和调试配置文件

(6)Peripherals 菜单和命令见表 2-6。

表 2-6　Peripherals 菜单和命令

菜　单	工具条	描　述
Reset CPU		复位 CPU
I/O-Ports	—	打开片上外围器件的设置对话框
Interrupt	—	打开中断观察窗口
Serial	—	打开串行口观察窗口
Timer	—	打开定时/计数器观察窗口
A/D Converter	—	A/D 转换器
D/A Converter	—	D/A 转换器
I^2C Controller	—	I^2C 控制器
CAN Controller	—	CAN 控制器
Watchdog	—	“看门狗”

(7)Tool 菜单和命令见表 2-7。

表 2-7　Tool 菜单和命令

菜　单	描　述
Setup PC-Lint	配置 Gimpel Software 的 PC-Lint 程序
Lint	用 PC-Lint 处理当前编辑的文件
Lint all C Source Files	用 PC-Lint 处理项目中的 C 源代码文件
Setup Easy-Case	配置 Siemens 的 Easy-Case 程序
Start/Stop Easy-Case	运行/停止 Siemens 的 Easy-Case 程序
Show File (Line)	用 Easy-Case 处理当前编辑的文件
Customize Tools Menu	添加用户程序到工具菜单中

(8)SVCS 菜单见表 2-8。

表 2-8　SVCS 菜单

菜　单	描　述
Configure Version Control	配置软件版本控制系统的命令

(9)Window 菜单见表 2-9。

表 2-9　Window 菜单

菜　单	描　述
Cascade	以互相重叠的形式排列文件窗口
Tile Horizontally	以不互相重叠的形式水平排列文件窗口
Tile Vertically	以不互相重叠的形式垂直排列文件窗口
Arrange Icons	排列主框架底部的图标
Split	把当前的文件窗口分割为几个
Close All	关闭所有窗口

(10)Help 菜单见表 2-10。

表 2-10　Help 菜单

菜　单	描　述
Help topics	打开在线帮助
About μVision	显示版本信息和许可证信息

2)Keil μVision3 中的调试器

Keil μVision3 中集成了一种新型调试器——Debug,它可以进行纯软件模拟仿真和硬件目标板在线仿真,使用之前应该先对其进行适当配置。单击 Project 菜单,选择 Options for Target 选项,弹出 Options for Target 对话框,切换到如图 2-2 所示 Debug 选项卡。

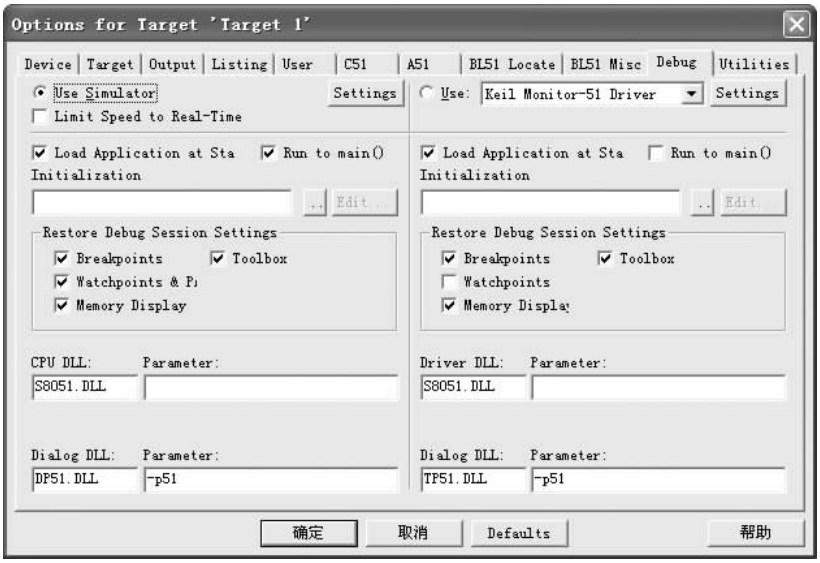


图 2-2　Debug 选项卡

在该选项卡中选中单选按钮 Use Simulator，可采用软件模拟方式进行仿真。这种模式下，可以在没有任何单片机硬件的情况下仅用一台计算机实现对单片机应用程序的仿真模拟。选中 Use 下拉列表框前的单选按钮，单击下拉按钮可选择不同器件对其仿真，例如，选择 Intel 8052 可仿真 8052 单片机内部定时器 T2，选择 Philips 80C552 可仿真 80C552 单片机内部 A/D 转换器的功能，选择 Monitor-51 Driver 可采用硬件目标板方式进行仿真。选择硬件目标板方式进行仿真时，要购买硬件仿真器，还要安装 Keil 公司提供的专用监控程序 Monitor-51，有了这两项后可实现对硬件目标系统的在线仿真。单击 Setting 按钮，弹出如图 2-3 所示对话框。单击 Port 下拉列表框中下拉按钮选择计算机的串口名，单击 Baudrate 下拉列表框中下拉按钮选择需要的通信波特率。对话框中间有四个复选框，是设定调试过程中数据的缓存选项，选中时可以加快数据在计算机屏幕上的显示速度，若希望直接观察单片机内部定时/计数器、I/O 端口各引脚电平状况以及外部扩展端口实际数据的变化，则不要选中这些复选框。选中 Serial Interrupt 复选框可以在运行程序过程中通过 Debug 调试器的 Stop 按钮或计算机键盘的 Esc 键停止程序的运行，但这时程序中不能使用单片机的串行口，同时不能禁止特殊功能寄存器 IE 中的 EA 位。

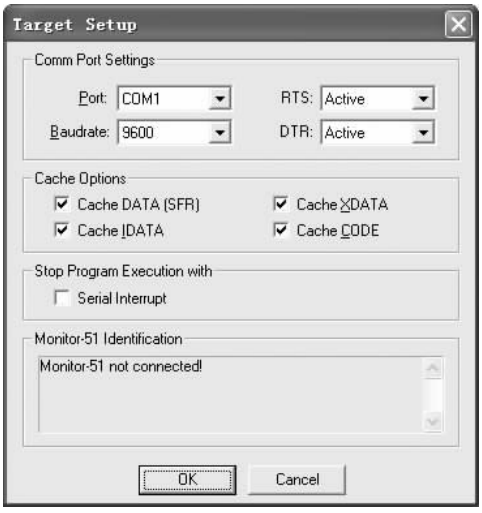


图 2-3　Target Setup 对话框

选中 Debug 选项卡中的 Load Application at Startup 复选框，在启动 Debug 时将自动装入程序，选中 Ran to main( )复选框，程序将从复位入口运行到 main( )函数处，通常这两个复选框都需要选中以便于调试。在 Initialization 文本框内可以输入一个带路径的初始化文

件名，该文件的内容为 Debug 的各种调试命令。可以在启动调试时一次执行。单击 Edit 按钮可以在编辑窗口打开初始化文件进行编辑。四个复选框 Breakpoints、Watchpoints&PA、Memery Display 和 Toolbox 分别用于在启动 Debug 时自动恢复上次调试过程中所设置的断点、观察点，性能分析器，存储器及工具盒的显示状态，如果希望启动 Debug 时能够使用在编辑源程序文件时就设置的断点，应该选中这些复选框。CPU DLL 文本框、Driver DLL 文本框、Dialog DLL 文本框和 Parameter 文本框是根据项目配置时从器件库中所选择的单片机 CPU 器件，由 Keil μVision3 自动设置的内部驱动程序及参数，一般不要轻易改动。

Debug 选项配置完成之后，打开或建立新项目并编译通过后，选择 Debug 菜单中的 Start/Stop Debug Session 选项，即可启动 Debug。启动 Debug 后，项目窗口分配如图 2-4 所示：寄存器窗口用于显示程序调试过程中单片机内部寄存器状态的变化情况；主调试窗口用于显示源程序，窗口左边的箭头指向当前程序语句，每执行一条语句箭头会自动向下移动，便于观察程序当前执行点，如果项目中包含有多个程序文件，执行过程中将自动切换到不同文件显示；命令窗口用于键入各种调试命令；存储器窗口用于显示程序调试过程中单片机的存储器状态；观察窗口用于显示局部变量和观察点的状态。此外，在主调试窗口位置还可以显示反汇编窗口、串行窗口以及性能分析窗口，通过选择 View 菜单中的相应选项，或单击相应工具按钮，可以切换窗口。下面具体介绍各窗口的操作。

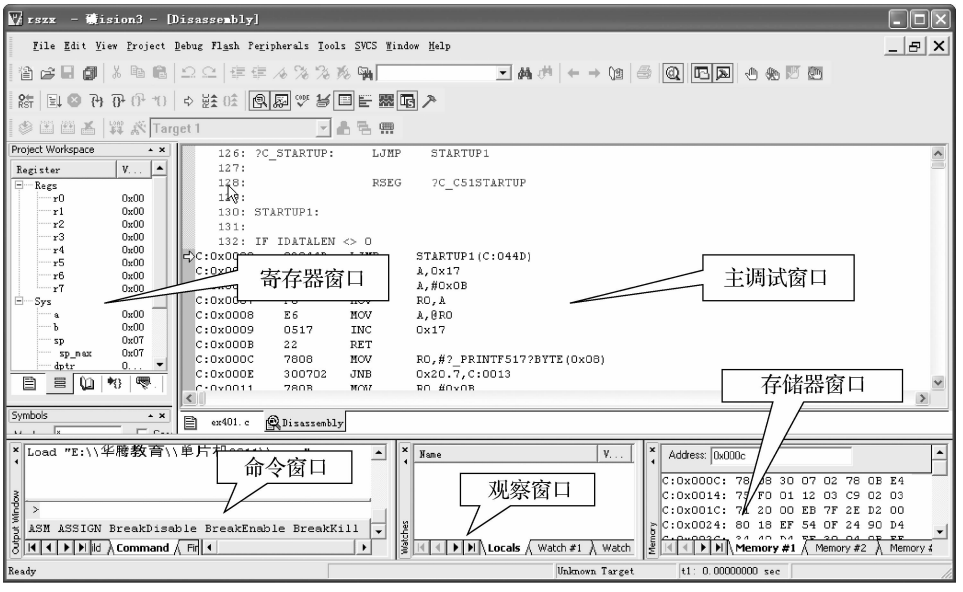


图 2-4　Keil μVision3 项目窗口分配

(1)命令窗口。调试状态下，单击 View 菜单，选择 Output Window 选项，弹出输出窗口，单击 Command 标签，可出现如图 2-5 所示命令窗口。

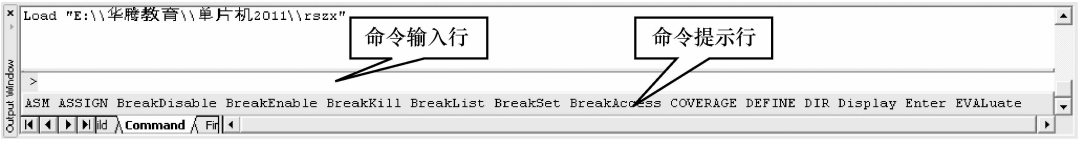


图 2-5　调试状态下命令窗口

窗口中有一个显示提示符“>”的命令输入行，其中可以输入各种命令字，如装入程序的目标文件，运行、设置观察点或断点等。在命令输入行内键入命令字并按 Enter 键，该命令将立即被执行，执行结果显示在输出窗口中，若命令输入错误，窗口中将显示出错信息。Keil μVision3 内部集成了一个方便实用的命令语法产生器，在输入命令时自动将所有可选的命令字、命令需要的参数等显示在窗口下边的命令提示行内。按 TAB 键可滚动显示提示行上的内容，键入提示行上某个命令字中的大写字母可直接输入该命令。在命令输入时提示行上的显示内容会自动减少，只剩下一个命令字时，键入空格即可输入该命令字，输入命令字后提示行上还将显示该命令所要求的各种参数，以避免出错。Keil μVision3 允许在输入命令字时带有 C 语言格式的注释。命令行可通过按键来改变命令字的输入，各按键功能见表 2-11。

表 2-11　按键功能

按　键	功　能
Enter	执行所输入的命令
Backspace	删除光标前面的一个字符
Del	删除光标处的一个字符
Esc	中止命令输入并开始新的命令输入行
Home	置光标于输入行首
End	置光标于输入行尾
Tab	滚动显示帮助行上的信息
←	光标左移一格
→	光标右移一格
↑	重复输入命令
↓	重复输入命令

(2)反汇编窗口。单击 View 菜单，选择 Disassembly Window 选项，弹出如图 2-6 所示的反汇编窗口，窗口中显示已经装入软件中的程序的汇编指令、反汇编代码及其地址。

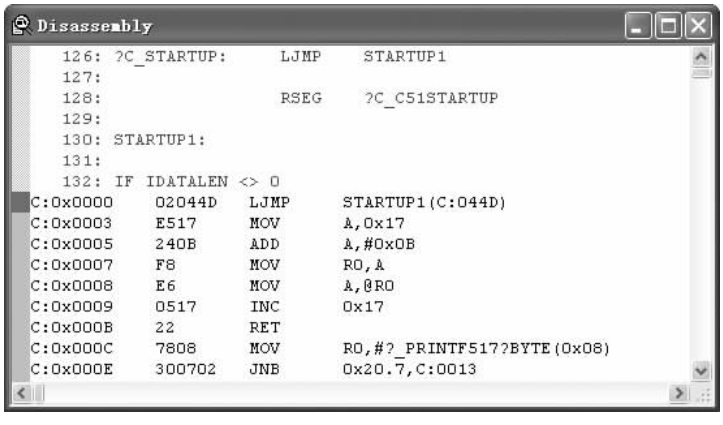


图 2-6　反汇编窗口

当采用单步或断点方式运行程序时，反汇编窗口的显示内容会随指令的执行而滚动。在反汇编窗口中右击可弹出如图 2-7 所示快捷菜单。

①快捷菜单第一栏。该菜单第一栏中共有三个选项，用于选择窗口内反汇编内容的显示方式。选择 Mixed Mode 选项，系统采用高级语言与汇编语言混合方式显示；选择 Assembly Mode 选项，系统采用汇编语言方式显示；选择 Inline Assembly 选项，系统用于程序调试中的“在线汇编”。使用“在线汇编”时，先将光标定位在反汇编窗口中的希望行上，再选择该选项弹出如图 2-8 所示对话框，其中 Current Instruction 文本框显示的是指定行当前的汇编指令，Current Assembly Address 文本框显示的是当前指令行的地址，在 Enter New Instruction 文本框中可以输入汇编语言指令，完成后按 Enter 键，新输入的指令将取代指定行上原来的指令，实现“在线汇编”功能。

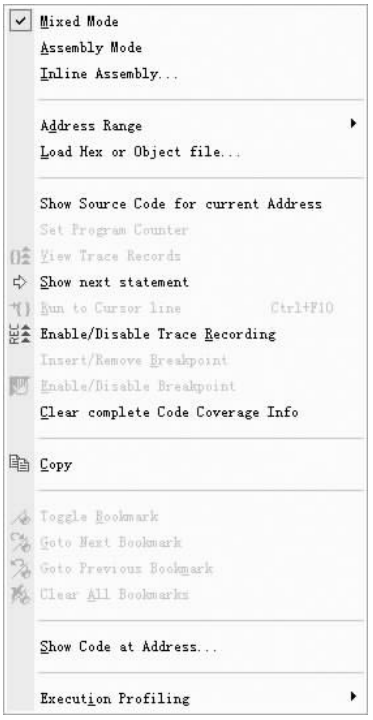


图 2-7　反汇编窗口快捷菜单

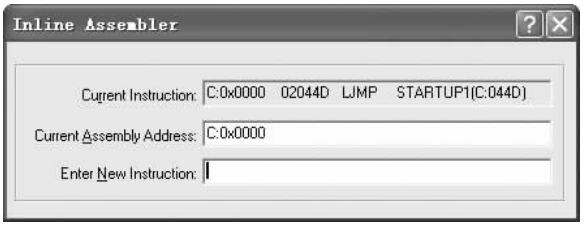


图 2-8　Inline Assembler 对话框

②快捷菜单第二栏。快捷菜单第二栏共有两个选项，Address Range 选项用于显示程序的地址范围；Load Hex or Object file 选项用于重新将 Hex 或 Object 文件装入到 Keil μVision3 中进行调试。

③快捷菜单第三栏。快捷菜单第三栏共有九个选项：Show Source Code for current Address 选项用于显示源代码和目前地址；Set Program Counter 选项用于设置程序计数器；View Trace Records 选项用于在反汇编窗口显示指令执行的历史记录，该选项只有在 Enable/Disable Trace Recording 选项被选中并且已经执行过程序指令的情况下才能起作用，这时反汇编窗口将以不同颜色显示出已经被执行过指令的历史记录，将光标向上移动可以查看更多历史记录，寄存器窗口的 Regs 选项卡中的显示内容也随之发生变化；Show next statement 选项用于显示下一条将被执行的指令；Run to Cursor line 选项用于将程序执行到

当前光标所在的那一行，使用方法是先将光标定位在需要暂停的程序上，然后选择该选项，程序会执行到指定行暂停；Insert/Remove Breakpoint 选项用于插入或删除程序执行时的断点，使用方法是先将光标定位在希望插入断点的程序行上，然后选择该选项，将在选定行插入一个断点，对于已经存在的断点，选择该选项可删除选定的断点；Enable/Disable Breakpoint 选项用于激活或禁止已经存在的断点；Clear complete Code Coverage Info 选项用于清除代码覆盖信息。

④快捷菜单第四栏。快捷菜单第四栏只有一个 Copy 选项，用于复制反汇编窗口中的内容。具体使用方法是：先选定反汇编窗口中需要复制的内容，右击选择该选项即可将选定的内容复制到剪贴板中，然后再将其粘贴到文件中。这一功能对于保存“在线汇编”时临时修改了的程序文件特别有用，因为“在线汇编”仅在调试程序时对装入软件中的汇编代码起作用，对源文件并未进行修改，退出调试之后，“在线汇编”的作用将不复存在。

⑤快捷菜单第五栏。快捷菜单第五栏用于文本标记操作，其中各个选项与菜单中“文件标记”选项的功能相同。

⑥快捷菜单倒数第二栏的 Show Code at Address 选项用于显示指定地址处的程序代码，选择该选项时反汇编窗口将立即切换到指定地址处的程序代码窗口。

(3)观察窗口。View 菜单的 Watch & Call Stack Window 选项用于调试状态下观察窗口的显示/隐藏切换。观察窗口有四个标签，分别是 Locals、Watch＃1、Watch＃2 和 Call Stack，图 2-9 所示为 Locals 窗口，窗口中显示程序调试过程中当前局部变量的使用情况。单击 Watch＃1 标签可切换到 Watch＃1 窗口显示程序中已经设置了的观察点在调试过程中的当前值。在 Locals 窗口或 Watch＃1 窗口中右击可改变局部变量或观察点的值，按十六进制(hex)或十进制(decimal)方式显示。单击 Call Stack 标签可切换到 Call Stack 窗口显示程序执行过程中对子程序的调用情况。

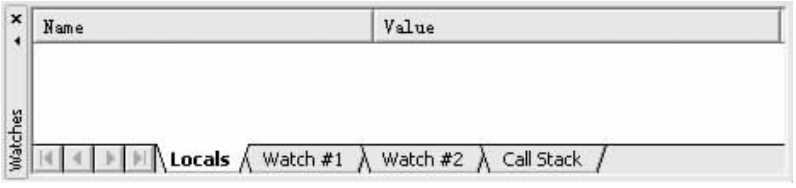


图 2-9　Locals 窗口

(4)存储器窗口。View 菜单的 Memory Window 选项用于系统存储器空间的显示/隐藏切换，选择该选项，弹出如图 2-10 所示存储器窗口。在 Address 文本框中输入存储器地址，将立即显示对应存储器空间的内容。输入地址时要指定存储器类型(C、D、I、X 等)。窗口有 Memory ＃1、Memory ＃2、Memory ＃3 和 Memory ＃4 四个标签，每个标签可指定不同的地址空间。

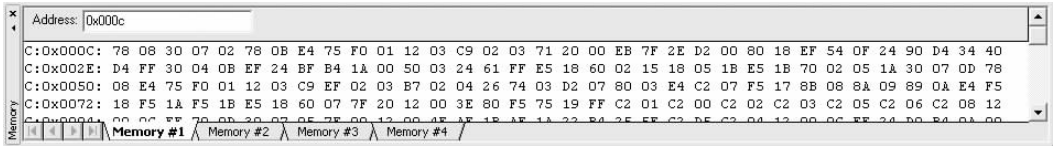


图 2-10　存储器窗口

(5)代码覆盖窗口。View 菜单中的 Code Coverage Window 选项用于代码覆盖窗口的显示/隐藏切换。选择该选项,弹出如图 2-11 所示代码覆盖性能分析窗口。性能分析窗口中显示的是指定程序模块的代码执行情况。在 Current 文本框内输入指定的程序模块名,该模块中各函数包含的指令条数及其已经被执行指令的百分数将会在窗口中显示出来。Update 按钮用于对显示值进行更新,Reset 按钮用于复位被执行指令的百分数。

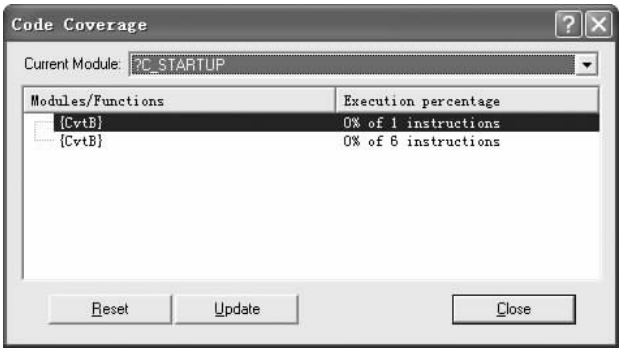


图 2-11　代码覆盖窗口

(6)性能分析窗口。View 菜单中的 Performance Analyzer Window 选项用于性能分析窗口的显示/隐藏切换,选择该选项,弹出如图 2-12 所示性能分析窗口。性能分析窗口用于显示指定程序模块被调用的次数及执行时间,分析结果以棒状图形式显示在窗口中,通过窗口内的一个运行性能统计标尺很容易了解该程序模块的运行性能。窗口中有六个文本框:min 文本框指定程序模块所消耗的最小时间;max 文本框指定程序模块所消耗的最大时间;avg 文本框指定程序模块所消耗的平均时间;total 文本框指定程序模块所消耗的总时间;% 文本框指定程序模块所消耗的时间占全部运行时间的百分数;count 文本框指定程序模块被调用的次数。

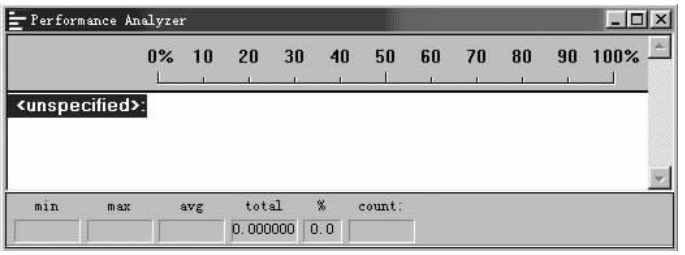


图 2-12　性能分析窗口

在性能分析窗口中右击弹出快捷菜单,共四个选项,其中,Reset PA 选项用于复位所有程序模块的性能分析值;Active PA 选项用于启动/停止性能分析;Update Times 选项用于更新程序执行时间;Setup PA 选项用于设置性能分析窗口的内容。选择 Setup PA 选项,弹出如图 2-13 所示的对话框。其中,Current PA Ranges 文本框显示的是当前已经设定的性能分析程序模块,单击 Kill All 按钮将删除已设定的全部模块,也可以选中其中某一模块再单击 Kill Selected 按钮删除该模块,在 Define Performance Analyzer 文本框中输入需要进行分析的程序模块名,再单击 Define 按钮可将其设定为一个性能分析程序模块,Function

Symbols 文本框中显示了当前所有可用的程序模块，将鼠标指向其中某个模块名并双击，选定的程序模块名将立即出现在 Define Performance Analyzer 文本框中，再单击 Define 按钮即可将其设定为一个性能分析程序模块。

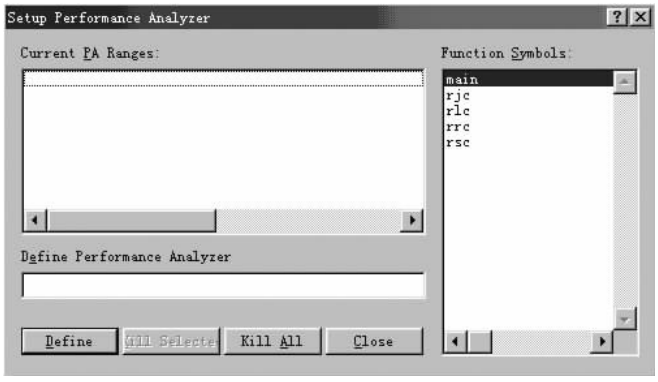


图 2-13　Setup Performance Analyzer 对话框

(7)符号窗口。View 菜单中的 Symbol Window 选项用于符号窗口的显示/隐藏切换，选择该选项，弹出如图 2-14 所示符号窗口，窗口中显示的是当前程序中的各种符号。在 Mask 文本框中输入希望显示的符号名，对应的符号信息将立即显示在窗口中，选中 Case Sensitive 复选框可区分大小写。

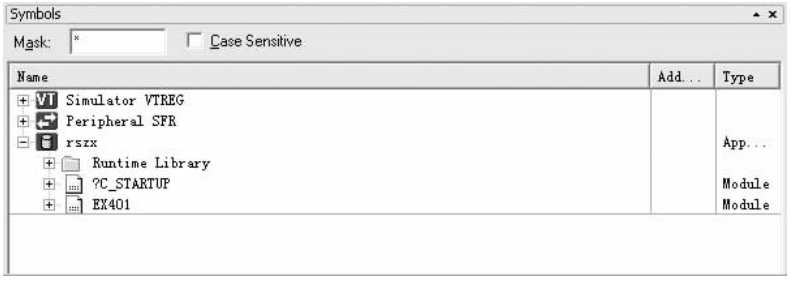


图 2-14　符号窗口

(8)串行窗口。View 菜单中的 Serial Window #1(Serial Window #2)选项用于串行窗口 1(2)的显示/隐藏切换。串行窗口在进行调试程序时十分有用，如果程序中调用了 C51 的库函数 scanf( )和 printf( )，则必须利用该窗口来完成 scanf( )函数的输入操作，printf( )函数的输出结果也将显示在该窗口中。利用串行窗口可在程序仿真调试过程中实现人机对话，或者对程序运行结果进行实时显示。在串行窗口中右击弹出快捷菜单，可按需要将窗口内容以十六进制或二进制显示，也可随时清除显示内容。串行窗口中可保持最近 8 KB 串行输入/输出数据，并可进行翻滚显示。

(9)工具盒。View 菜单中的 Toolbox 选项用于工具盒窗口的显示/隐藏切换，选择该选项，弹出如图 2-15 所示的窗口，单击 Update Windows 按钮将对当前显示内容进行更新。

图 2-15　工具盒窗口

在该窗口中，可以根据需要自定义其他命令按钮。自定义按钮可以通过在命令窗口中选择 Define Button 选项来实现，其格式为

```
Define Button "button_label","button_command"
```

其中，button_label 是按钮名，button_command 必须是一个有效的 Keil μVision3 命令。

例 2-1 定义工具盒按钮。

```
>Define Button "go to main"                    //定义 "go to main"按钮
>Define Button "clr dptr" ,"dptr=0"            //定义"clr dptr"按钮
```

进行正确定义后，对应的按钮将立即出现在工具盒窗口中，每个自定义命令按钮前自动冠以一个编号，以便于删除该按钮。随着自定义按钮数目的增加，工具盒窗口将自动增大。在命令窗口中选择 Kill Button 选项可以删除自定义命令按钮，预定义按钮 Updata Windows 前面没有编号，因此它是不能被删除的。

例 2-2 删除工具盒自定义按钮。

```
>Kill Button 1                                 //删除编号为1的自定义按钮
>Kill Button 2                                 //删除编号为2的自定义按钮
```

(10)周期更新。View 菜单中的 Periodic Window Update 选项用于对各窗口的显示内容进行周期性更新。在进行程序调试时，若希望从观察窗口中看到某个变量值的改变情况，必须选择该选项。

3)Debug 菜单功能

在 Debug 中可以进行两种类型的代码调试：带调试信息的源程序代码调试和 Hex 代码调试。前者允许调试过程中显示高级语言源程序语句，后者仅能显示基本汇编语言指令。在成功完成项目编译连接之后，通过 Debug 菜单进入程序调试状态，在调试状态下仍可通过调试主窗口进行源程序的编辑修改，这也是 Debug 的一大特点，可以根据当前调试结果修改源程序，但修改后不能立即进行调试，要先退出当前调试状态，重新编译连接成为新的目标代码再次装入之后才能调试。

Debug 菜单中的 Start/Stop Debug Session 选项用于启动/停止调试功能，启动之前应先确定是采用软件模拟仿真还是采用硬件目标板仿真。启动调试功能之后，项目窗口自动切换到 Regs 窗口，显示当前单片机内部各寄存器的状态，寄存器状态将随着程序语句或指令的执行而变化。

Debug 菜单第二栏的各选项用于控制目标代码的执行方式。选择 Go 选项，程序从当前地址处开始全速运行，遇到断点或选择 Stop Running 选项时停止。Step 选项用于单步执行，在高级语言显示方式下选择一次该选项执行一条 C 语言语句，在汇编语言显示方式下则执行一条指令，遇到函数调用(高级语言方式)或子程序调用(汇编语言方式)语句时，将跟踪进入函数或子程序中执行。选择 Step Over 选项，程序从当前地址处开始执行一条语句，对于函数或子程序调用语句不跟踪进入被调用函数，而是将整个函数或子程序与调用语句一起一次执行。在调用函数或子程序过程中，选择 Step Out of current Function 选项，函数或子程序从当前地址处开始执行并返回到调用函数或子程序的下一条语句，该选项对于非函数或子程序以及采用硬件目标板仿真时无效。选择 Run to Cursor Line 选项，程序从当前地址处开始执行到光标所在行。Stop Running 选项用于停止运行程序。

Debug 菜单第三栏各选项用于程序中的断点管理。断点功能对于程序的仿真调试是十分重要的，它可在某个特定地址或是满足某种特定条件下暂停程序，以便观察、了解程序的运行状况，查找或排除错误。选择 Breakpoints 选项将弹出如图 2-16 所示的 Breakpoints 对

话框。Current Breakpoints 文本框用于显示当前已经设置的断点列表，Keil μVision3 自动为每个断点分配一个内部序号，序号前面有一个复选框，选中时使能该断点，否则禁止该断点，禁止断点不同于删除断点，它仅仅暂时禁止断点发挥作用，需要时可以重新激活该断点，让它重新发挥作用；Expression 文本框用于键入断点表达式；Count 文本框用于键入断点通过次数，例如，当 Count 的值等于 2 时，表示在第二次运行到该断点时停止程序运行或执行规定的命令；Command 文本框用于键入当程序执行到断点时需要执行的命令；Access 栏用于规定断点的存取类型，选中 Read 复选框表示读，选中 Write 复选框表示写，同时选中这两个复选框表示读写；Size 栏用于规定存取断点的长度，选中 Bytes 复选框时，按断点表达式的值从第一个地址开始计算其字节长度。

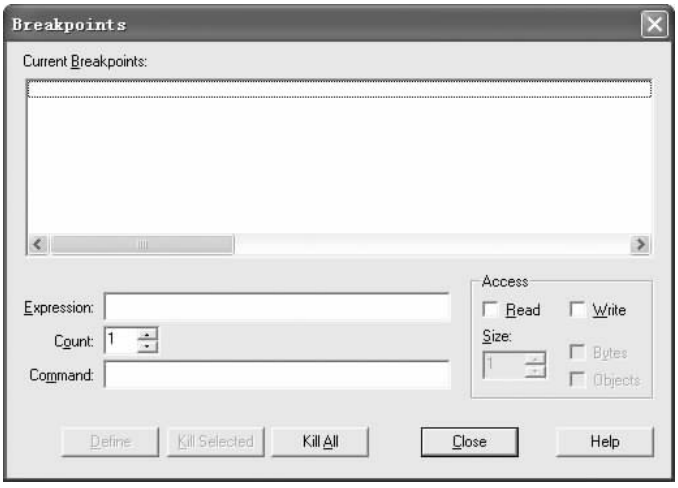


图 2-16　Breakpoints 对话框

常用断点有三种：执行断点（execution breakpoint）、条件断点（conditional breakpoint）和存取断点（access breakpoint）。它们在使用中各有优缺点。若断点表达式为一个特定代码地址，则为执行断点，程序每当运行到该地址时被暂停，如函数地址或语句行号。执行断点必须设置在正确的代码地址处，对于某些代码地址只能设置一次断点，不能重复定义，利用执行断点可以检查程序流程是否正确。若断点表达式不是代码地址，即为条件断点。对于条件断点，每一条指令执行结束后重新计算其逻辑表达式，若计算逻辑结果为假（“0”值），继续执行程序，若计算逻辑结果为真（非“0”值），则暂停执行程序。利用条件断点可以检查程序中的一些特殊错误，但可能会降低程序的执行速度。带有存取类型（Read、Write）标志的断点为存取断点，用于捕捉对于非法存储器地址的存取操作。

双击 Current Breakpoints 文本框中某个断点，与该断点有关的信息将在窗口下边的相关栏中显示出来。选中某个断点后单击 Kill Selected 按钮，可以立即删除该断点，单击 Kill All 按钮将删除全部断点。设置新断点时，可直接在 Expression 文本框内输入断点表达式，在 Command 文本框和 Count 文本框内输入有效的 C51 命令串和计数值（如果 Count 栏为空，则断点计数值默认为 1），对于存取断点还需要在 Access 栏选定相应的存取属性，然后单击 Define 按钮，新断点即设置完成。当 Command 文本框中输入了命令串时，该命令串将在程序到达断点时被立即执行，而程序并不在断点处暂停。若需要在断点处暂停程序，必须在

命令串中增加表达式“_BREAK_=1”。断点命令串为可选项，并非每个断点都需要命令串，因此Command文本框可以为空，此时每当程序执行到断点时都将会暂停。下面举几个设置断点的例子。

例2-3 设置执行断点。

Expression文本框：main

Command文本框：printf ("main has been reached\n"),_BREAK_=1

Count文本框：1

在程序中标号“main”处设置一个执行断点，当程序执行到标号“main”时将自动执行命令串“printf("main has been reached\n")”，同时在命令窗口内输出命令串执行结果。由于命令串中存在表达式“_BREAK_=1”，因此程序在断点处被暂停。

例2-4 设置执行断点。

Expression文本框：\MEASURE\110

Command文本框：

Count文本框：1000

在程序模块MEASURE的行号110处设置一个执行断点，计数值为1 000。当程序在第1 000次执行到该断点地址时将被暂停，断点起作用之后计数值将变为1。

例2-5 设置条件断点。

Expression文本框：(sindex > 0x0a && sindex<0x25)||sindex==0x133

Command文本框：eval sindex

Count文本框：1

当满足条件“0x0a<sindex<0x25或sindex==0x133”时程序将被暂停，并执行命令串“eval sindex”，命令执行结果显示在命令窗口中，执行完命令串后继续运行程序。

例2-6 设置条件断点。

Expression文本框：$==time0 && sindex > 5

Command文本框：

Count文本框：1

当程序执行到函数timer0()并且当变量sindex>5时，暂停执行程序。

设置条件断点时，其表达式必须表示一个存储器地址和存储器类型，并且要遵循两条规则：表达式结果必须具有唯一的存储器类型；只允许使用&、&&、<、<=、>、>=、==、!=等运算符。

例2-7 设置存取断点。

Expression文本框：sindex && sindex==0x133

Command文本框：

Access栏：选中write复选框

Count文本框：1

当程序向变量sindex进行写入，且当变量sindex的值为0x133时暂停执行。

除了可以通过上述断点设置窗口来设置断点外，还可以直接通过Debug菜单来设置断点。进入调试状态后，选择Debug菜单中的Insert/Remove Breakpoint选项，可在编辑窗口当前光标所在文件行上插入或删除一个断点；选择Debug菜单中的Enable/Disable Breakpoint

选项，可激活/禁止当前光标所指向的一个断点；选择 Debug 菜单中的 Disable All Breakpoints 选项，将禁止所有已经设置的断点；选择 Debug 菜单中的 Kill All Breakpoints 将删除所有已经设置的断点；选择 Debug 菜单中的 Show Next Statement 选项将在编辑窗口中显示下一条将要被执行的语句。

Debug 菜单中的 Enable/Disable Trace Recording选项用于激活/禁止程序调试时跟踪指令执行的历史记录，在该选项被激活并且已经执行过程序的情况下，选择 View Trace Records 选项，可以从汇编窗口查看源程序指令执行的历史记录。

Debug 菜单中的 Memory Map 选项用于进行软件模拟仿真时设置存储器空间映像，选择该选项，弹出如图 2-17 所示 Memory Map 对话框。Current Mapped 文本框用于显示所有已被设置的存储器空间映像，选中其中一项并单击 Kill Selected Range 按钮将删除选定的存储器空间映像，删除后不能对该部分存储器进行操作，否则将在命令窗口显示 access violation 错误；Map Range 文本框用于希望设置存储器空间映像的地址范围，格式为

字母前缀：起始地址，字母前缀：终止地址

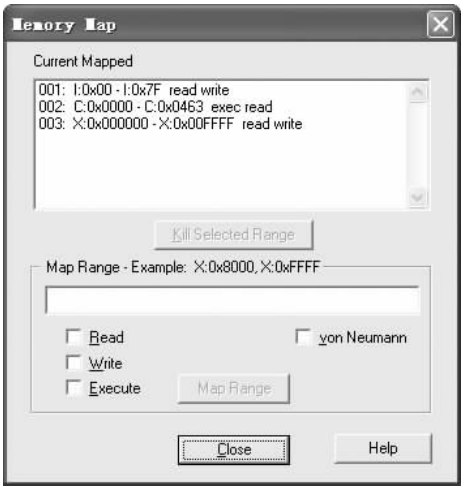


图 2-17　Memory Map 对话框

Keil μVision3 软件的存储器空间映像可高达 16 MB，采用以下字母来区分不同的存储器空间映像，见表 2-12。

表 2-12　存储器空间映像设置

字母前缀	存储器空间映像
I(D)	256 B 内部数据存储器(DATA、IDATA)空间，地址范围为 0x00～0xFF
X	64 KB 外部数据存储器(XDATA)空间，地址范围为 0x0000～0xFFFF 16 MB 外部数据存储器(HDATA)空间，地址范围为 0x000000～0xFFFFFF
C	64 KB 代码存储器(CODE)空间，地址范围为 0x0000～0xFFFF 16 MB 代码存储器(ECODE)空间，地址范围为 0x000000～0xFFFFFF
B0～B31	分组代码空间，B0 地址范围为：0x0000～0xFFFF；B31 地址范围为：0x0000～0xFFFF

除了可以设置以上标准单片机存储器空间映像之外，Keil μVision3 还可以设置四个自定义存储空间映像，字母前缀为 S、T、U、V，地址范围为 0x0000～0xFFFF。

存储器空间映像还带有 Read、Write、Execute 或 von Neumann 属性，带有属性的存储器空间映像只能进行与之相应的操作。内部数据存储空间具有默认的 Read、Write 属性。von Neumann 属性用于将外部数据存储器与代码存储器相重叠，此时对于任何外部数据存储器(XDATA)空间的存取操作，同时发生在代码存储器(CODE)空间。需要注意的是，设置 von Neumann 属性的存储器空间映像应同时具有 Read 和 Write 属性，代码存储器空间不能被设置为 von Neumann 属性。单击 Map Range 按钮即完成存储器空间映像设置。

虽然 Keil μVision3 软件支持 16 MB 存储器空间映像，但通常只需要对程序所使用的存储器空间范围进行设置，存储器空间映像设置得过大会降低 Keil μVision3 的运行速度。启动调试功能并装入目标程序之后，Keil μVision3 软件将自动按当前项目创建时对存储器的要求设置存储器空间映像。一般来说，自动设置的存储器空间映像已经可以满足程序调试要求，只有在一些特殊场合才需要重新进行设置。

例 2-8 设置内部数据存储器空间映像。

Map Range 文本框输入“I:0x00,I:0xFF”，选中 Read 复选框和 Write 复选框。

例 2-9 将外部数据存储器空间 0x8000～0xFFFF 与代码存储器空间 0x8000～0xFFFF 相重叠。

Map Range 文本框输入“I:0x8000,I:0xFFFF”，选中 Read 复选框、Write 复选框和 von Neumann 复选框。

例 2-9 中，在外部数据存储器地址 0x8000～0xFFFF 内进行读写操作与在代码存储器地址 0x8000～0xFFFF 内进行读写操作的效果相同。

Debug 菜单中的 Performance Analyzer 选项，用于设定需要进行性能分析的某段程序的地址范围(简称 PA 范围)。

Debug 菜单中的 Inline Assembly 选项，用于程序调试过程中的在线汇编。

Debug 菜单中的 Function Editor(Open Ini File)选项，用于编辑或创建 C51 的初始化文件，初始化文件中可以包含各种 C51 命令及调试函数。选择该选项，弹出如图 2-18 所示窗口，窗口中的 Open 按钮用于打开已有的初始化文件；New 按钮用于创建一个新的初始化文件；Save 按钮和 Save as 按钮用于保存当前编辑的初始化文件；Compile 按钮用于编译并加载当前编辑的初始化文件，如果发生错误，将在 Compile Errors 文本框中显示相应的错误号，同时光标自动跳转到第一个错误所在的行上，以利于编辑修改。对于初始化文件中包含的 C51 命令，编译时将会自动执行。

4)Peripherals 菜单功能

目前 MCS-51 系列单片机已有 400 多个品种和型号，不同型号具有不同的外围集成功能，Keil μVision3 通过内部集成器件库实现对各种单片机外围集成功能的模拟仿真，在调试状态下可以通过 Peripherals 菜单来观察仿真结果。Peripherals 菜单的选项内容会根据选用器件库中不同器件而有所变化，图 2-19 所示为选用 8052 单片机器件后的 Peripherals 菜单。

图 2-18　初始化文件编辑窗口

图 2-19　Peripherals 菜单

(1)Peripherals 菜单中 Reset CPU 选项用于对模拟仿真的单片机进行复位。

(2)Peripherals 菜单中的 Interrupt 选项用于显示单片机中断系统状态,选择 Interrupt 选项后将弹出如图 2-20 所示的窗口。通过对 Selected Interrupt 栏各中断标志位复选框选中或不选中,可以选中不同的中断源,从而实现对单片机中断系统的仿真。对于具有多个中断源的单片机,如 8052 单片机,除了几个基本中断源之外,还可以对监视定时器等其他中断源进行模拟仿真。

(3)Peripherals 菜单中的 I/O-Ports 选项用于仿真单片机的并行 I/O 端口 P0 口～P3 口,选中 Port1 后将弹出如图 2-21 所示的窗口,其中 P1 文本框显示单片机 P1 口寄存器状态,Bits 文本框显示 P1 口的 8 只引脚状态,仿真时各位的状态可根据需要进行修改。对于具有多个 I/O 端口的单片机,如 8052 单片机,P0 口～P3 口都是一样的,其他 I/O 端口略有不同。

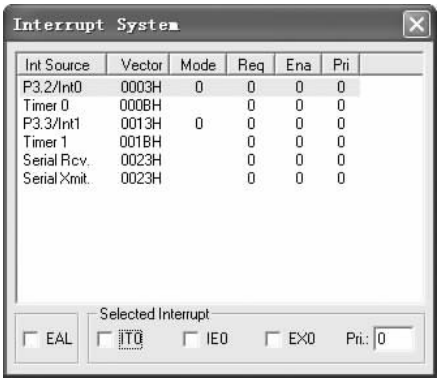


图 2-20　“中断系统”对话框

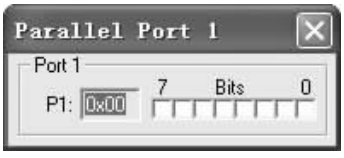


图 2-21　单片机 P1 口仿真

(4)Peripherals 菜单中的 Serial 选项用于仿真单片机的串行口,单击该选项将弹出如图 2-22 所示的窗口。Mode 下拉列表框用于选择串行口的工作方式,单击下拉按钮可选择不同的工作方式。选定工作方式后,相应特殊工作寄存器 SCON 和 SBUF 的控制字将显示在相应文本框中。通过对 SM2、REN、TB8、RB8、TI 和 RI 复选框的置位和复位操作(选中或不选中),很容易实现对单片机内部串行口的仿真。Baudrate 文本框用于显示串行口的工作波特率,选中 SMOD 复选框时将使波特率加倍,IRQ 栏用于显示串行口的发送和接收中

断标志。

(5)Peripherals 菜单中的 Timer 选项用于仿真单片机内部定时/计数器，选择 Timer0 选项后弹出如图 2-23 所示窗口。窗口中的 Mode 下拉列表框用于选择工作方式，可选择定时器或计数器方式，单击下拉按钮可进行选择。选定工作方式后，相应特殊工作寄存器 TCON 和 TMOD 的控制字也显示在相应文本框中，TH0 文本框和 TL0 文本框用于显示计数初值，T0 Pin 复选框和 TF0 复选框用于显示定时/计数器的溢出状态。窗口的 Status 文本框用于显示和控制定时/计数器的工作状态(Run 或 Stop)。TR0、GATE 和 INT0＃复选框是启动控制位，通过对这些状态位的置位和复位操作(选中或不选中)对单片机内部定时/计数器仿真。对于具有多个定时/计数器的单片机，如 8052 单片机，其 T0 和 T1 与 MCS-51 系列单片机是一样的，监视定时器等其他状态和控制略有不同。

图 2-22　串行口窗口

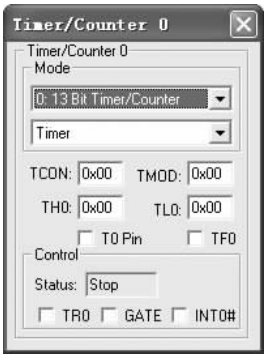


图 2-23　Timer0 窗口

2. 调试方法

启动 Keil μVision3，单击 Project→New→μVision Project，如图 2-24 所示，建立一个项目。

图 2-24　建立项目

从弹出的窗口中，选择要保存项目的路径，并输入项目文件名“EX201”，然后单击“保存”按钮，如图 2-25 所示。

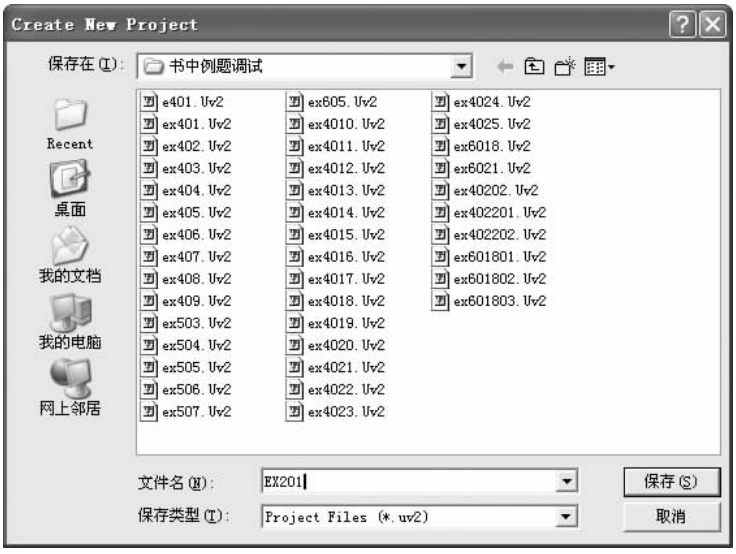


图 2-25　“保存”窗口

弹出一个选择 CPU 型号的对话框，可以根据所使用的单片机来选择，如图 2-26 所示。选定 CPU 型号之后从窗口中可以看到对这个单片机的基本说明，然后单击“确定”按钮。

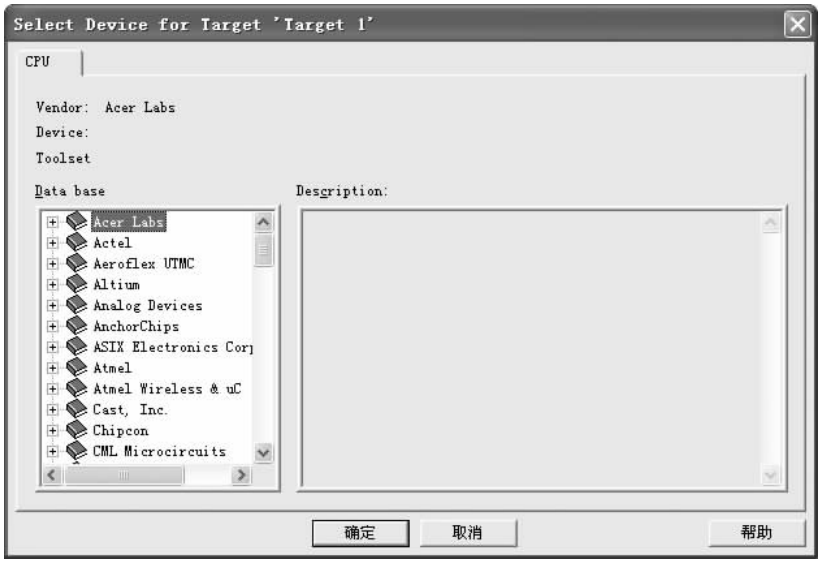


图 2-26　选定 CPU 型号

选择 File 菜单中的 New 选项，创建程序文件，如图 2-27 所示。

图 2-27　新建文件

在弹出的“编辑”窗口中输入程序，如图 2-28 所示。程序输入完成后，单击汇编工具按钮，系统对程序进行编译。编译时，系统逐行检查语法是否有错误，有错时编辑窗口最下面的信息窗口中会列出所有错误。双击错误行，光标跳转到对应的程序行，可对错误程序进行修改，必须一一修改正确，直到没有错误为止。

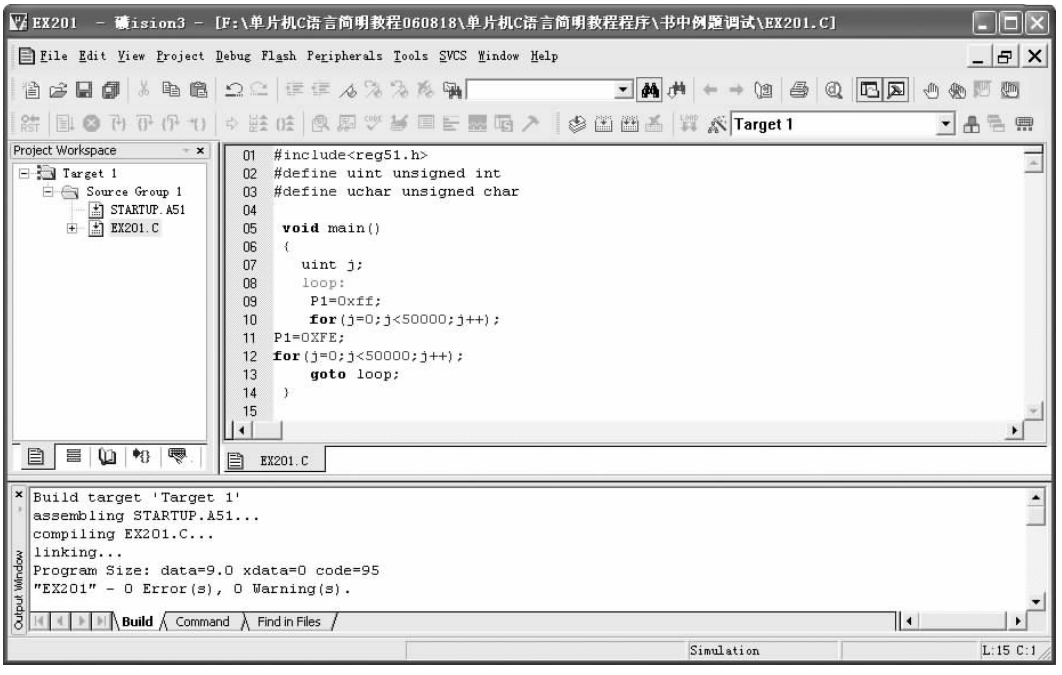


图 2-28　“编辑”窗口

选择 File 菜单中的 Save as 选项，如图 2-29 所示。从弹出的窗口中，选择要保存程序文件的路径，并输入程序文件名“EX201.C”，然后单击“保存”按钮，如图 2-30 所示。

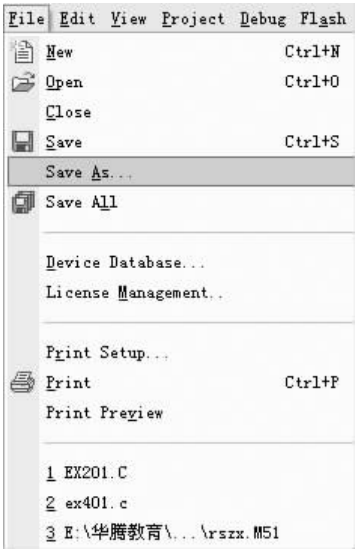


图 2-29　File 菜单中选择 Save As

图 2-30　Save As 窗口

将刚才创建的程序文件添加到项目中去。先单击 Target 1 前面的“+”，展开里面的内容“Source Group 1”，然后右击“Source Group 1”弹出一个快捷菜单，选择快捷菜单中的 Add Files to Group 'source Group 1'选项，如图 2-31 所示。

从弹出的窗口中选择刚才保存的文件“EX201.C”，并单击 Add 按钮，将文件添加到项目中，如图 2-32 所示。“STARTUP.A51”和“reg.h”自动加入文件目录下。

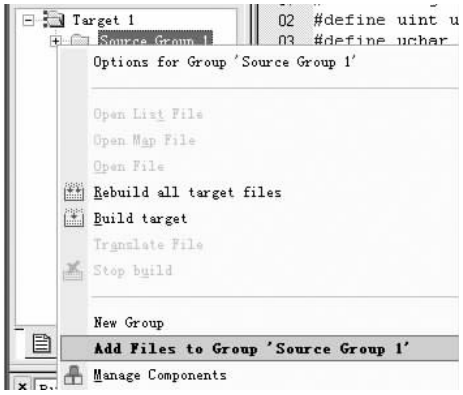


图 2-31　添加文件

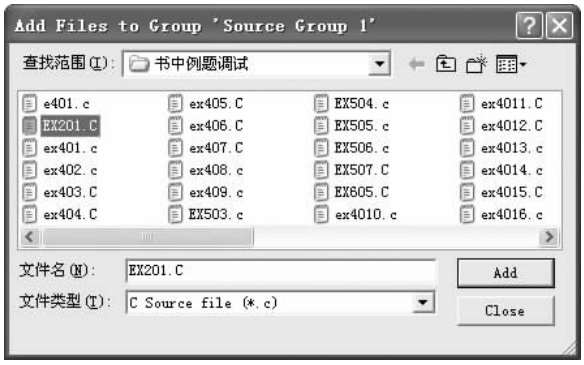


图 2-32　Add Files 窗口

程序文件添加完毕后，右击“Target 1”弹出快捷菜单，选择菜单中的 Options for Target 'Target 1'选项，如图 2-33 所示。

从弹出的 Options for Target 'Target 1'窗口中分别选择 Target 选项卡、Output 选项卡、C51 选项卡、BL51 Locate 选项卡和 Debug 选项卡，设置各选项卡中参数和选项，如图 2-34～图 2-38 所示。最后单击“确定”按钮，完成各项设置。

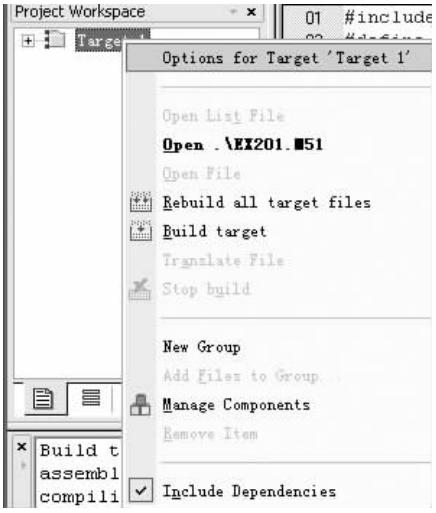


图 2-33　打开工程文件快捷菜单

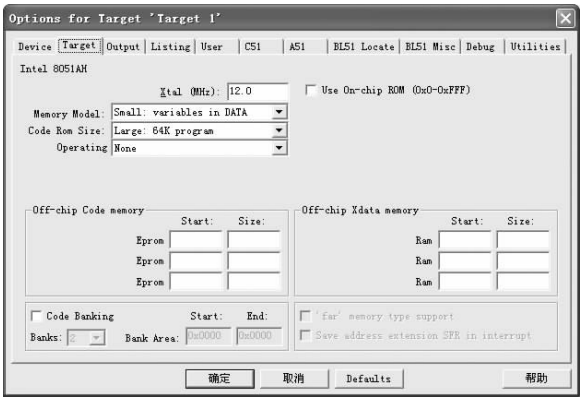


图 2-34　Target 选项卡

图 2-35　Output 选项卡

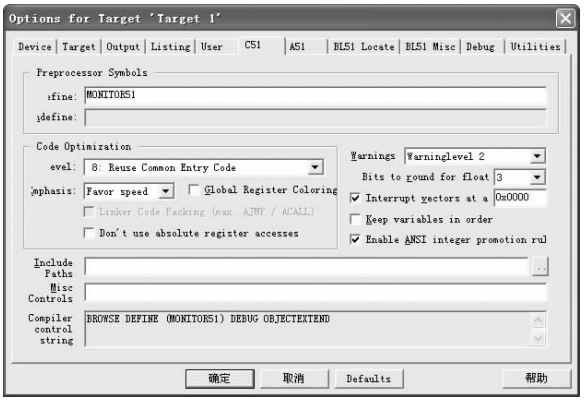


图 2-36　C51 选项卡

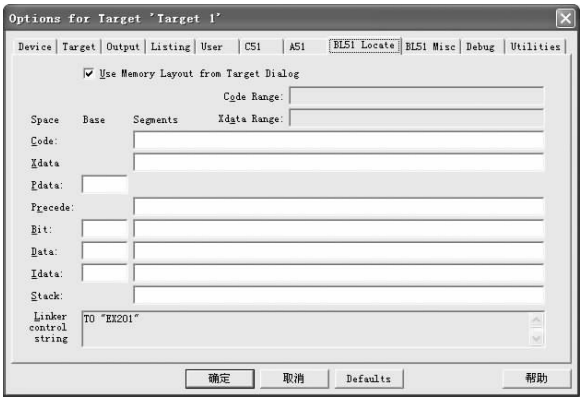


图 2-37　BL51 Locate 选项卡

图 2-38　Debug 选项卡

选择 Project 菜单中的 Rebuild all target files 选项，建立项目文件，如图 2-39 所示。

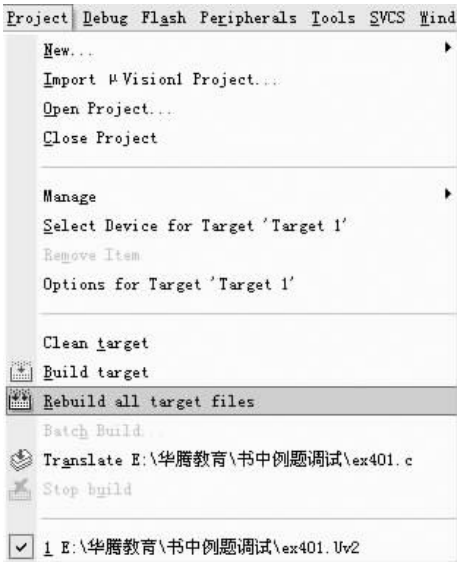


图 2-39　“编译”菜单

对项目中的程序文件进行编译连接，并生成与项目文件同名的可执行代码及用于 EPROM 编程的 HEX 文件。如果没有错误，Keil μVision3 环境将如图 2-40 所示。

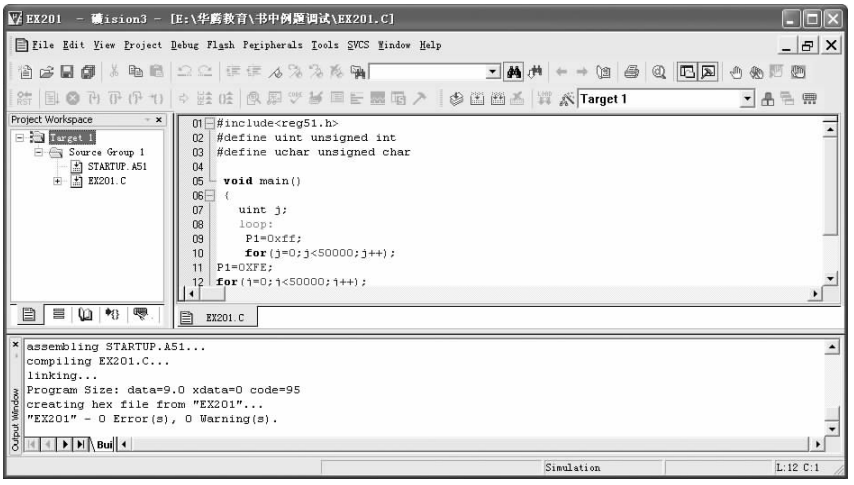


图 2-40　编译信息窗口

选择 Debug 菜单中的 Start/Stop Debug Session 选项，开始进入调试状态，如图 2-41 所示。

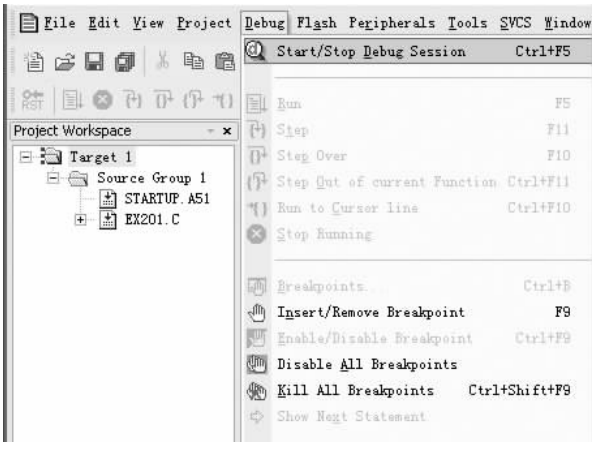


图 2-41　Debug 菜单

进入调试状态后，Keil μVision3 开发环境将显示联机状态及监控程序版本号，如图 2-42 所示。

图 2-42　版本号显示窗口

调试时，一般是先用全速命令运行一次，看程序功能是否能实现。若有错，要反复修改、调试程序。

二、硬件仿真

Keil C51 软件只能仿真编写的 C51 程序是否正确，不能仿真单片机接口芯片。硬件仿真需要使用 Proteus 软件，下面具体讨论 Proteus 软件的使用方法。

进入 Proteus 仿真环境。在主窗口建立新文件后，选择 Library 菜单中的 Pick Device/Symbol 选项，如图 2-43 所示。

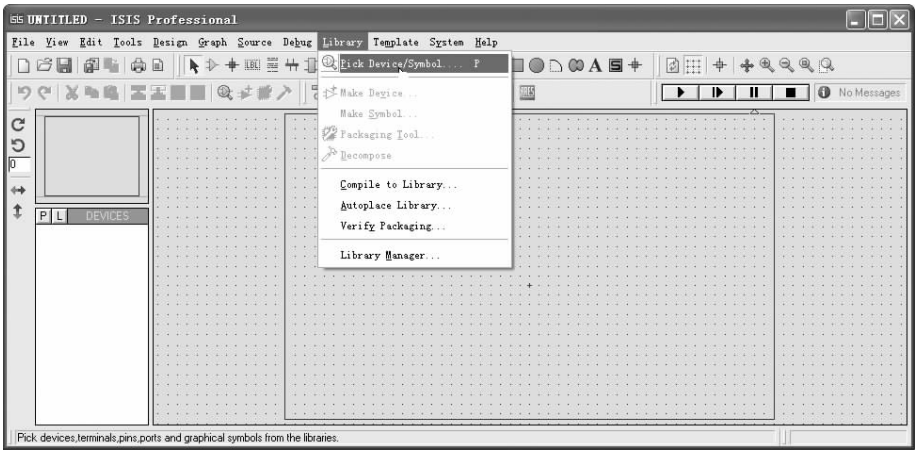


图 2-43　Library 菜单

在弹出窗口中的 Keywords 文本框处输入“89C51”，搜索结果显示在 Results 文本框中，选择第一项，单击 OK 按钮，如图 2-44 所示。

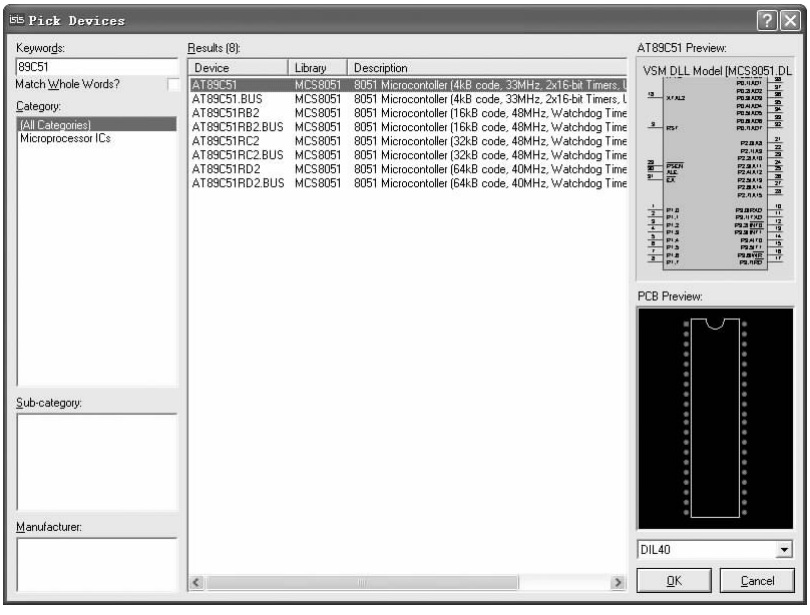


图 2-44　选择元器件

返回主窗口，单击窗口将 AT89C51 单片机放在主窗口中，如图 2-45 所示。参照放置单片机方法，可放置其他元器件，作出电路原理图。

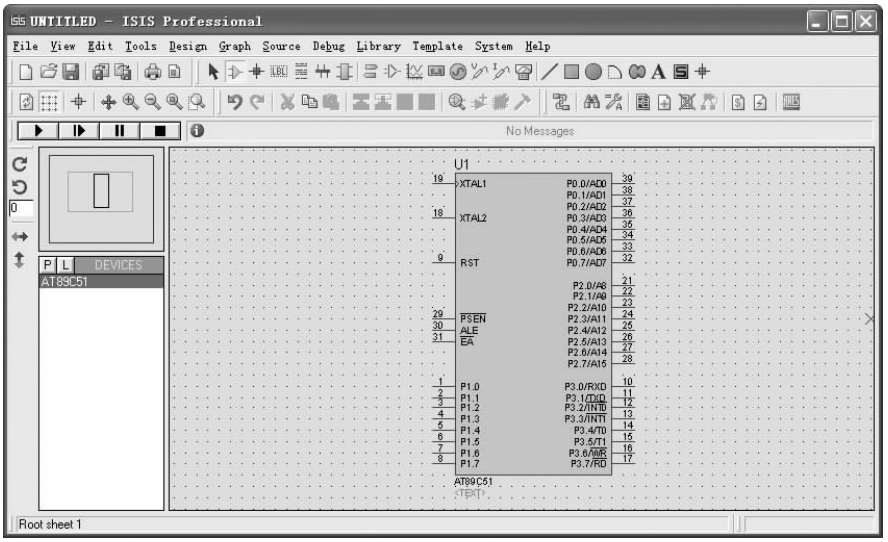


图 2-45　放置元器件

作图时在每个元件的旁边会显示灰色的文字“<TEXT>”，为了使电路图清晰，可以取消此文字显示。双击此文字，打开如图 2-46 所示对话框。切换到 Style 选项卡，不选中 Visible 复选框及其后面的 Follow Global 复选框，再单击 OK 按钮即可。

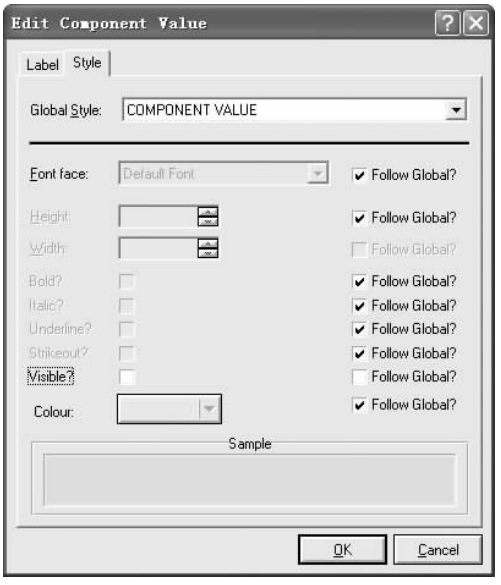


图 2-46　Style 选项卡

作出电路原理图后可装载程序文件，装载程序文件有两种方法。

(1)装载源程序到 Proteus 软件中。先选择 Source 菜单中的 Add/Remove Source Files 选项装载源程序，然后汇编程序，汇编成功后，再仿真。使用该方法时首先要进行两个软件

的联合使用设置。

(2)装载源程序到芯片中。双击“AT89C51”,出现如图 2-47 所示的对话框。单击 Program Files 文本框右侧工具按钮,出现文件浏览对话框,找到“EX201.hex”文件,单击“确定”按钮,完成添加。在 Clock Frequency 文本框中把频率改为 8 MHz,单击 OK 按钮退出。

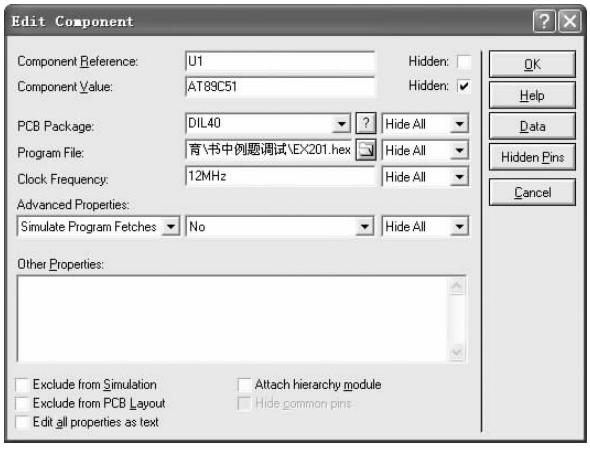


图 2-47　添加程序文件

单击工具按钮或按快捷键 F12 可进行仿真运行。

三、在线仿真

前面已经讲了硬件仿真软件的使用方法,但仿真软件有其局限性,它只能仿真特殊功能寄存器、内部 00H～7FH 单元和外部 0000H～FFFFH 单元,若扩展了 8155 芯片、8255 芯片、ADC0809 芯片、DAC0832 芯片之类的外部接口,就不能用软件仿真,此外,它不能进行硬件系统的诊断和实时仿真。要解决这些问题,可用硬件在线仿真器。在线仿真器种类很多,基本上归为两大类:一类是专用仿真芯片组成的仿真器;一类是一些非专业生产厂家制作的使用 Keil C51 界面的简易仿真器。这里重点讨论启东市微机应用研究所研制生产的 QTH 系列单片机仿真器。

1. QTH 系列单片机仿真器概述

QTH 系列单片机仿真器按其通讯接口可分为串口仿真器、并口仿真器和 USB 口仿真器三类,其软件系统和 QTH 系列实验系统使用同一个界面,是全新的具有 Visual C 界面风格的窗口,支持 Windows 98/ME/2000/XP/NT 操作系统,使用起来比较方便。硬件系统除通信电缆不同外,总体结构一样,如图 2-48 所示。

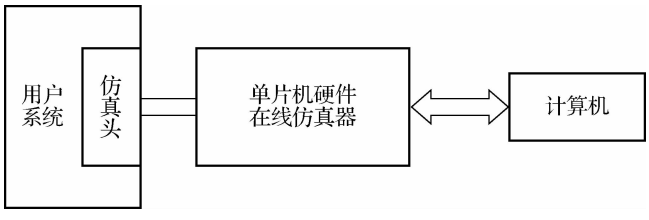


图 2-48　通用单片机仿真开发系统

先将要开发的产品硬件做好，检查无误后，把单片机和外接的程序存储芯片（AT89C51单片机没有外接的程序存储芯片）拔掉，然后用仿真器自带的仿真插头一头接在线仿真器，一头通过仿真头插入单片机插座（注意不要插反），再连接好电源线和通信线，最后装好与仿真器配套的软件，就可以对硬件进行在线仿真了。在整个仿真开发系统中，最关键的是仿真器。仿真器有以下六个功能。

（1）仿真功能。在线仿真时，仿真器中的单片机完整地出借给用户系统，不占用用户系统单片机的任何资源，使联机仿真和脱机运行时的环境（工作程序、使用的资源和地址空间）完全一致。芯片上的CPU、RAM、SFR、定时/计数器、中断源、I/O端口以及外部可扩充的程序存储器和数据存储器的地址空间，应允许用户系统充分、自由地使用，不应受到任何限制，方便用户系统根据单片机固有的资源特性进行硬件和软件的设计，实现完全的一次性仿真。

（2）模拟功能，特别是程序存储器模拟功能。在初始阶段，要编写程序，程序暂时存放在开发系统RAM存储器内，以便于在调试过程中对程序进行模拟和修改。开发系统所能出借的作为用户系统程序存储器的RAM称为仿真RAM。开发系统中，仿真RAM容量和地址映射应和用户系统完全一致。对于MCS-51系列单片机开发系统，最多应能出借64 KB的仿真RAM，地址为0000H～FFFFH，并保持原有的复位入口和中断入口地址不变。

（3）源程序编辑功能。仿真器允许在源程序窗口进行各种编辑操作，其功能和使用方法类似于计算机的WORD文档编辑功能。

（4）汇编功能。仿真器可以自动识别源程序，并根据其扩展名性质选择编译器对程序进行汇编，同时生成目标文件和列表文件。

（5）反汇编功能。仿真器可以对目标板上的程序进行反汇编，也可对调试运行的指令进行反汇编；还可以直接调入指导扩展名为hex或obj的文件进行反汇编调试。对程序进行反汇编所得到的程序清单可以打印或存入主机磁盘，这为调试和分析单片机产品中的软件提供了一个有效的手段。

（6）调试功能。仿真器能控制实验系统以单步、断点、跟踪、连续等方式运行程序。在运行过程中，可打开CPU窗口显示出单片机的基本状态；也可打开观察窗口根据需要选择想要观察的变量进行观察或修改其数据；还可以打开存储器窗口显示数据存储器或程序存储器的内容，使程序运行的结果显示在屏幕上。

2. QTH系列单片机仿真器的使用方法

QTH系列单片机仿真器有各种接口，除了通讯方式不同外，其他软硬件的操作方式都一样。通讯方式不同的仿真器由其系统软件自动识别，无须再进行端口设置。将仿真器与PC用配套的专用电缆线相连，安装好配套软件并设置好参数后，开始进行仿真。

QTH集成开发环境提供了两种方式开发应用程序：一种不使用QTH集成开发环境项目管理方式，即对源程序文件直接进行汇编/连接，兼容传统开发习惯；一种使用QTH集成开发环境项目管理方式，可进行多模块、混合语言编程，同样也适合单模块程序的开发。

无论是进行单模块还是多模块的程序开发，都建议使用项目管理方式开发应用程序。下面分别对这两种开发方式进行介绍。

1）不使用项目管理方式开发应用程序

不使用QTH集成开发环境项目管理方式只能对单模块方式下的应用程序开发，具有

很大的局限性。不使用项目管理方式开发应用程序的步骤如下。

(1)关闭当前项目文件。不使用 QTH 集成开发环境项目管理方式开发应用程序,必须关闭已经打开的项目。因为打开项目文件后,QTH 集成开发环境默认所有编译/汇编、产生代码的过程都是对项目或项目所包含的文件进行的。关闭当前项目文件的操作方式是:单击“项目管理”菜单,选择“关闭当前项目”命令。

(2)在“文件”菜单中创建或打开应用程序。单模块方式下,要创建一个新的程序文件,可使用三种方式:单击“文件”菜单,选择“新建”选项;单击工具栏中“新建”工具按钮;使用 Ctrl+N 组合键。单模块方式下,要打开一个已经存在的程序文件,可使用三种方式:单击“文件”菜单,选择“打开”选项;单击工具栏中“打开”工具按钮;使用 Ctrl+O 组合键。

(3)编译/汇编。QTH 集成开发环境根据文件的扩展名自动对当前激活的文件选择调用外部编译器或汇编器,有三种操作方式:单击“项目”菜单,选择“编译当前文件”选项;单击工具栏中“编译”工具按钮;使用快捷键 F3。如果需要设置文件“编译/汇编”的命令参数,可以单击“项目管理”菜单选择“文件属性”选项进行设置。

(4)产生代码并装入仿真器调试。编译通过后还必须进行连接操作,可以直接对当前文件进行编译连接和装载操作。此命令自动地对修改过的源程序进行编译或汇编,然后连接所有的 obj 和 lib 文件,再装载代码到仿真器,完成调试程序所需的准备工作。装载完成后,调试器窗口调试工具条所有命令按钮变亮。装入仿真器调试有两种操作方式:单击“项目”菜单,选择“编译连接装载”选项;单击“项目”菜单,选择“装入”选项。

(5)选择进入在线仿真或模拟仿真。当仿真器连接正常后即可自动进入在线仿真状态;如果联机不正常,则仿真器会提示联机出错,并询问是否要进入模拟仿真,单击“是”按钮,进入模拟仿真状态。

经过以上五个步骤,屏幕上出现 QTH 集成开发环境的系统画面。

2)使用项目管理方式开发应用程序

使用 QTH 集成开发环境项目管理方式,可以对单模块和多模块方式应用程序进行开发。使用项目管理方式开发应用程序的步骤如下。

(1)新建项目。单击“项目”菜单,选择“新建项目”选项,仿真器打开一个“新建项目”对话框,在对话框中选择“立即加入模块文件”选项,选择文件添加到项目管理器中。QTH 集成开发环境的项目文件是按项目名称管理的,项目管理器内的项目名称不可以相同。在项目名称输入栏内,必须输入项目名称,并且项目名称不得超过 8 个字符,不可以使用汉字以及-、?、*、/等字符。

(2)加入模块文件。单击“项目”菜单,选择“加入模块文件”选项,在当前新建或打开的项目中添加源程序文件,必须把主模块放在第一栏中。

(3)打开项目。调试已经存在的项目,可以直接打开项目文件。打开项目的操作方式是:单击“项目”菜单,选择“打开项目”选项。

(4)设置项目属性。设置当前项目的编译及连接控制项属性有两种方法:单击“项目”菜单,选择“项目属性”选项;单击“设置”菜单,选择“项目属性”选项。

(5)其余步骤同不使用项目管理方式开发应用程序一样。

项目实践

用1片AT89C51单片机、1个LED和1个电阻组成LED闪动控制电路，用字节操作的方法编写控制由P1.0引脚控制的1个指示灯闪动(一亮一灭)的程序。程序设计完成后分别进行软件仿真和硬件仿真。

控制外接于P1.0引脚的LED亮和暗时，就是在P1.0引脚输出一个方波，考虑到LED从亮到暗有延时效应和人眼的视觉暂留，亮的时间应大于0.5 s，这样才可以观察到LED有亮和暗的变化。参考程序如下。

```
#include<reg51.h>
#define uint unsigned int
#define uchar unsigned char
void main()
{
  uint j;
  loop:
    P1=0xff;
    for(j=0;j<50000;j++);
    P1=0XFE;
    for(j=0;j<50000;j++);
    goto loop;
}
```

在Keil C51软件中输入以上程序，单击Debug工具按钮，程序开始位置出现运行光标。打开P1口模拟图，每按一下F7，执行一条指令，如图2-49所示。重复以上调试方法，反复循环，程序调试成功后将程序汇编、固化到芯片中。

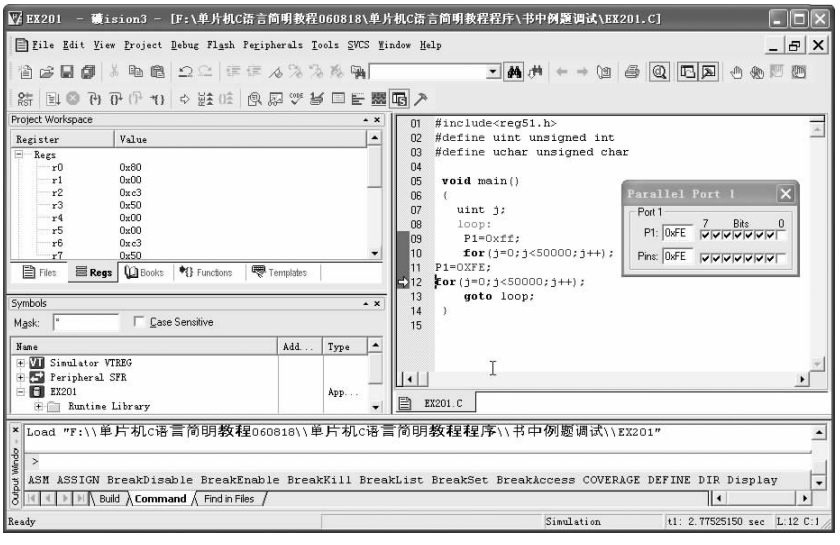


图2-49　程序调试窗口

在 Proteus 软件中输入如图 2-50 所示电路原理图，进行硬件仿真，仿真结果可看到接于 P1.0 引脚的 LED 一亮一灭闪动。

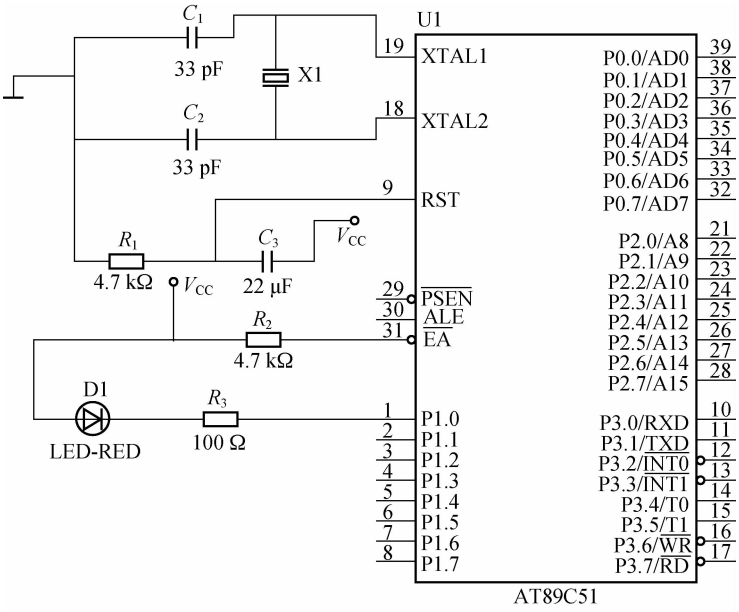


图 2-50　电路原理图

拓展知识

1. ISP 单片机

进行单片机的产品开发时，通常需要借助编程器将调试好的程序烧入单片机内部程序存储器中。另外，在开发过程中，程序每改动一次就要拔下电路板上的芯片，程序改动完成后再插上，也比较麻烦。随着单片机技术的发展，出现了可以在系统编程(ISP)的单片机。

1)ISP 单片机概述

ISP 单片机的开发不需要编程器，单片机可以直接焊接到电路板上，调试结束即成成品，甚至可以远程在线升级单片机中的程序。以 Winbond(华邦)公司生产的 W78E516B 单片机为例，其最大的优点是：带有 64 KB 的闪速存储器，还有独立的 4 KB ROM 专门用来固化在线仿真程序；与 MCS-51 系列单片机指令集完全兼容，只是片内 RAM 增加为 1 KB，多一个定时/计数器 2，多两个特殊功能寄存器和专对 4 KB ROM 区进行操作的指令。

2)ISP 单片机的使用方法

(1)设计好产品硬件，做一根通信线(一般是买一根标准串口通信线)，一端(A 端)与计算机相连，另一端的插头不用，直接将通信线的三根线分别与单片机的串口和地相连。

(2)在软件仿真系统中调试好程序，汇编生成 HEX 文件并命名保存。

(3)运行 ISP，主界面如图 2-51 所示。

(4)单击“打开文件”按钮，选择“H→B”选项，将 HEX 文件转成二进制文件。

(5)单击“连接”按钮，目标文件传到 64 KB 闪速存储器中，单击“复位”按钮即可运行

程序。

此方法不能仿真调试，只能是一次一次的固化后调试程序，程序有错只能在仿真软件中检查修改，修改后又调试，反复进行，直到成功为止。

图 2-51　ISP 主界面

2. 单片机网站

随着网络技术的普及和发展，为学习、使用、开发单片机提供了极大的方便。在网上可了解芯片的最新动态，查找、下载资料，借鉴他人的最好设计方案，进入网络论坛商讨技术问题，买到最低价位的芯片等。有以下几种类型的网站。

1)芯片生产厂家网站

全世界各大芯片厂家都有自己的网站。在这些网站上，可以了解到该厂生产的所有芯片以及各种芯片的技术资料、使用方法、开发实例等。网站一般都设有下载专区，可下载所需的技术资料和开发实例，为使用该芯片提供了极大的方便。

2)单片机销售网站

由于大的生产厂家一般不对外直接销售，而是使用分销商或代理商，代理又分为一级代理和二级代理，代理级别越高，销量越大，厂家给的价格也越低。代理商为了推销产品建立了自己的网站，这些网站只出售芯片，如 www.e-ic.com.cn、www.21ic.com.cn 等。

3)专业网站

专业网站以某专业为对象开设网站，如专门卖开发仿真器的网站，专门卖实验设备的网站等。

4)混合网站

混合网站既做芯片销售又做技术开发，还销售各种设备、开发器和编程器，如 www.zlg-mcu.com。这类网站的好处是，可讨论一些技术问题，在开发中遇到技术问题时可提供一些帮助。

5)学习网站

学习网站大多是各院校为了教学之用专门建立的,主要为学生提供教学服务,网站内有教材、教学课件、专题讨论、习题解答、开发实例、疑难问题解答等。如 www.gg.51dz.com。

复习思考题

1.在线仿真系统中单片机与计算机的通信线怎样连接?

2.在线仿真与仿真软件的主要区别在哪里?

项目三
单片机最小系统设计

知识目标

掌握最小系统的组成；

掌握最小系统的应用。

技能目标

掌握最小系统的软件仿真方法；

掌握最小系统的硬件仿真方法

项目描述

单片机裸机组成是学习单片机的第一步，任何大的单片机应用系统都是由裸机扩展而来的。裸机设计初学者最大的困惑就是感到无从下手，这是因为对单片机应用设计不甚了解。对于任何一类单片机都有使用说明书，说明书中都有裸机设计示例，只要按要求设计器件参数，做出电路原理图，一般情况下是可以运行成功的，除非芯片损坏。

相关知识

一、单片机最小系统概述

一个最小的单片机微机系统由三片集成块组成，它们是 CPU、8 位三态 D 锁存器 74LS373、ROM 或 RAM，习惯上将这三样称为老三件。当然，有了这三样，单片机还是不能工作，还要加上一个时钟电路和复位电路，由这些基本电路组成一个完整的最小系统，称为裸机，如图 3-1 所示。该电路可提供 P1 口和 P3 口作为用户的 I/O 端口，即常说的人机接口

通道。下面具体讨论芯片的使用方法。

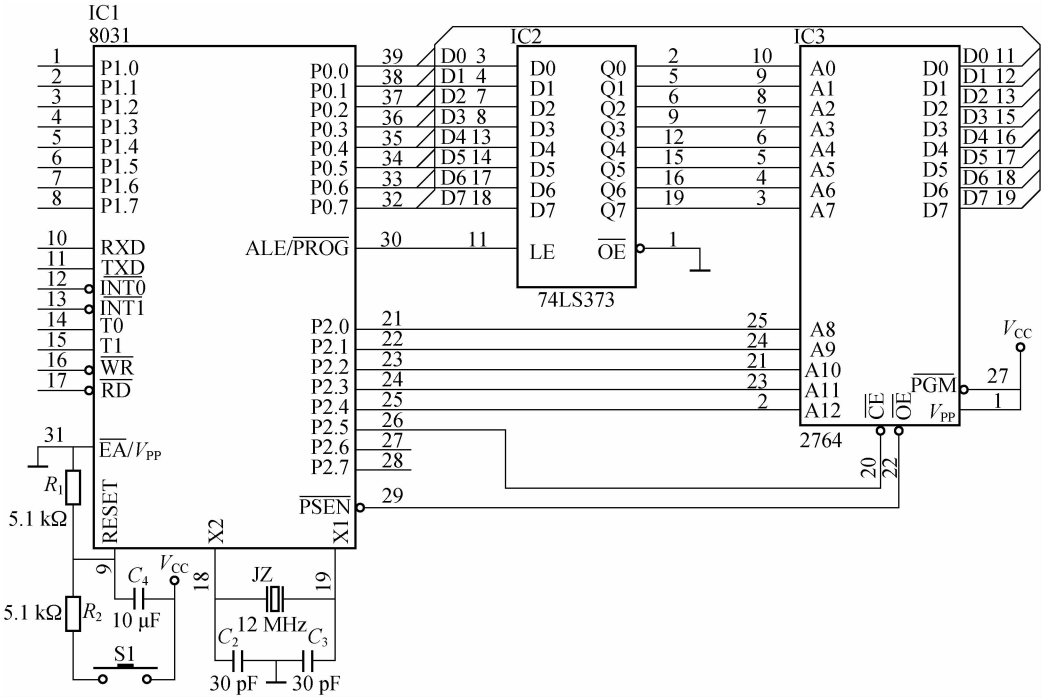
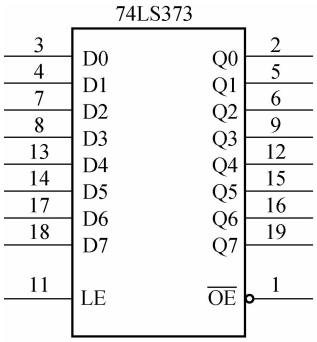


图 3-1　8031 单片机最小系统

1.8 位三态 D 锁存器 74LS373 的使用

一般的集成块生产厂家都提供全套集成块的使用说明书，说明书中主要包括该集成块的特点、逻辑图、引脚功能图、特性、电参数、工作原理和典型应用。74LS373 锁存器的引脚图和功能表如图 3-2 所示。



(a)引脚图

LE	D*n*	Q*n*	$\overline{OE}$
H	H	H	L
H	L	L	L
L	L	L	L
L	H	H	L
×	×	Z	H

(b)功能表

图 3-2　74LS373 锁存器引脚图和功能表

图 3-2(a)中，D*n* 为输入端；Q*n* 为输出端；$\overline{OE}$和 LE 为控制端。74LS373 锁存器如何工作由图 3-2(b)的功能表确定，表中 L 为低电平，H 为高电平，Z 为高阻抗(相当开路)，×为任意电平。一般将$\overline{OE}$引脚接低电平，LE 引脚接 8031 单片机的 ALE 引脚，就能正常工作。

2.2764 芯片的使用

2764 芯片是一块 8 KB 的 EPROM 程序存储器，有 28 只引脚，分成地址线、数据线和控制线。

3.最小系统的解释

1)分时使用的方法

(1)硬件连接。P0口一路直接与2764芯片的数据口相连,一路通过74LS373锁存器后与2764芯片的低8位地址线相连。在物理上将数据信号通道和地址信号通道分开。工作时与软件配合分时传送数据信号和地址信号。

(2)软件。程序在执行时一条一条地执行,在时间上也是分时的。

2)存储器容量的计算方法

存储器容量的计算方法为

$$2^{\text{地址线根数}}=\text{存储器容量}$$

从图3-1可知,2764芯片的地址总线为A0～A12,共计13根,所以2764芯片的存储容量为$2^{13}=2^{10}2^3=8$ KB。

3)片选地址的计算

将P2.5引脚接片选信号$\overline{CE}$,P2.6引脚和P2.7引脚接低电平,则有

	P2.7	P2.6	P2.5	P2.4	P2.3	P2.2	P2.1	P2.0	P0.7～P0.0	
0000	0	0	0	0	0	0	0	0	0	0
…										
1FFF	0	0	0	1	1	1	1	1	1	1

所以,存储器的地址范围是0000H～1FFFH。

可用AT89C51单片机来完成此电路的功能,如图3-3所示。

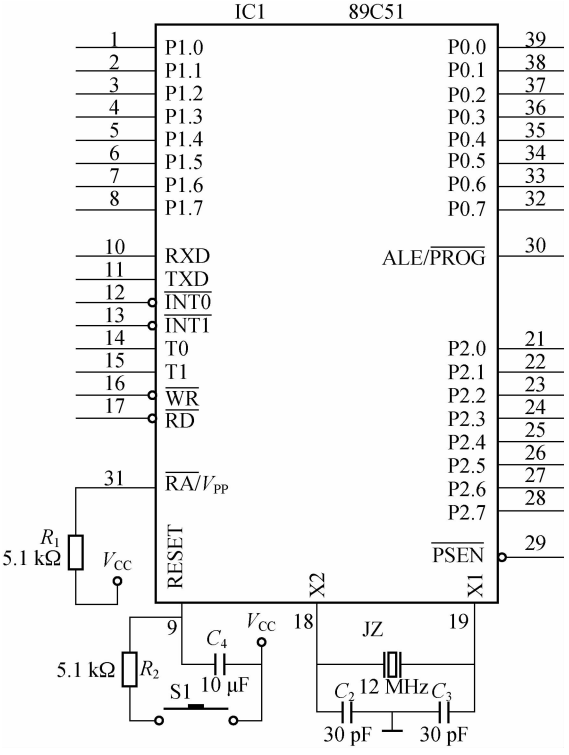


图3-3 AT89C51单片机最小系统

二、单片机最小系统的应用

1. 简单控制部件简介

最小系统只是单片机能工作的最低要求，它不能对外完成控制任务，也不能实现人机对话。作控制时需要执行部件，要进行人机对话还要一些 I/O 部件。常见的执行部件有继电器、电磁阀等，输入部件有开关、按钮、键盘、鼠标等，输出部件有指示灯、数码管、显示器等，下面只介绍几个简单的部件。

1）继电器

继电器是用低电压控制高电压的器件，它分为线圈、铁心、衔铁和触点，触点有常开触点、常闭触点之分。在开关特性上继电器有单刀单置、双刀单置、单刀双置、双刀双置、单刀多置和双刀多置之别。图 3-4 所示为继电器的电气符号，方框为线圈，圆圈为触点，直线为刀，图中只列了四种类型的继电器：图 3-4(a)所示为双刀双置，图 3-4(b)所示为双刀单置，图 3-4(c)所示为单刀单置，图 3-4(d)所示为单刀双置。

继电器的工作过程是：线圈得电时，常开触点闭合，常闭触点断开；线圈失电时，常开触点断开，常闭触点闭合。电路连接时，单片机的一个输出口接线圈的一端，线圈的另一端接符合线圈电压标准的电源。以单刀单置为例，将 220 V 相线断开接触点两端（相当于在相线上接一个开关），220 V 线上再接电器设备。当用软件控制单片机的该输出口为低电平时，线圈得电，常开触点闭合，电器设备工作（设定低电平工作）；用软件控制单片机的该输出口为高电平时，线圈失电，常开触点断开，电器设备停止工作（设定高电平停止）。

2）光耦

光耦在电路中起隔离作用，由光作为信号传递媒介（工具），将单片机和外部设备在电气上隔离。光耦分为三极管型光耦（又分带基极型和不带基极型）和可控硅型光耦（又分单向可控和双向可控）两种。图 3-5 所示为光耦的电气符号。

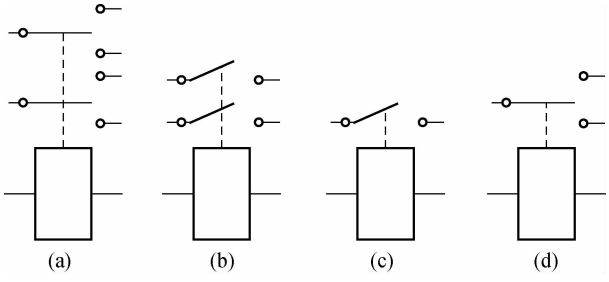


图 3-4　继电器符号

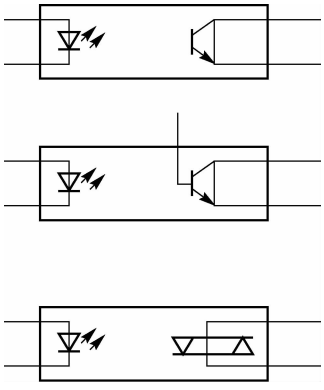


图 3-5　光耦符号

三极管型光耦的工作过程是：有电流通过内部发光管，发光管发光，所对应的内部三极管导通；无电流通过内部发光管，发光管不发光，所对应的内部三极管不导通（断开）。一般接法是内部发光管阳极接高电平（电源正极），与单片机同电源；阴极接单片机的某一输出口；内部三极管对外的两端接外部设备，这就将单片机和外部设备在电气上分隔开。当用软

件控制单片机的该输出口为低电平时，内部发光管发光，所对应的内部三极管导通，外部设备工作（设定低电平工作）；用软件控制单片机的该输出口线为高电平时，内部发光管不发光，所对应的内部三极管不导通，外部设备停止（设定高电平停止）。

3）指示灯

指示灯相当于一个二极管，加正向电压发光，反之不发光。一般接法是阳极接高电平（电源正极），阴极接单片机的某一输出口，当该输出口为低电平时，指示灯亮，该输出口为高电平时，指示灯不亮。这样只要编程控制单片机的该输出口，就可控制指示灯亮或灭。

有了以上知识，可设计一些实用电路，下面以彩灯控制器为例，全面讲解产品的设计过程、开发工具的使用方法和开发产品时要注意的问题。

2. 彩灯控制器的设计

最小系统最简单的应用就是在I/O端口上接彩灯，简单易学，在实验室就能做。下面讨论其设计过程。

1）硬件设计

最小系统是固定的，功能可一部分一部分地扩展。对于彩灯控制器，只要在I/O端口上加驱动器就行。因8031单片机每个端口带负载的能力有限，如果在P1口的8个引脚上同时直接接8个指示灯，则不能编程控制8个指示灯同时亮。要想8个指示灯能同时亮，要给每一个引脚加驱动，一般用同相放大器或反相放大器作为驱动器，若驱动电流还不够，应改用三极管作驱动器件。根据以上设计思路，可作如图3-6所示电路图，图中最小系统没变，只是加了8个发光二极管、8个电阻和8个同相放大器74LS244，电阻是限流电阻，调整阻值使发光管最亮，电流最小。

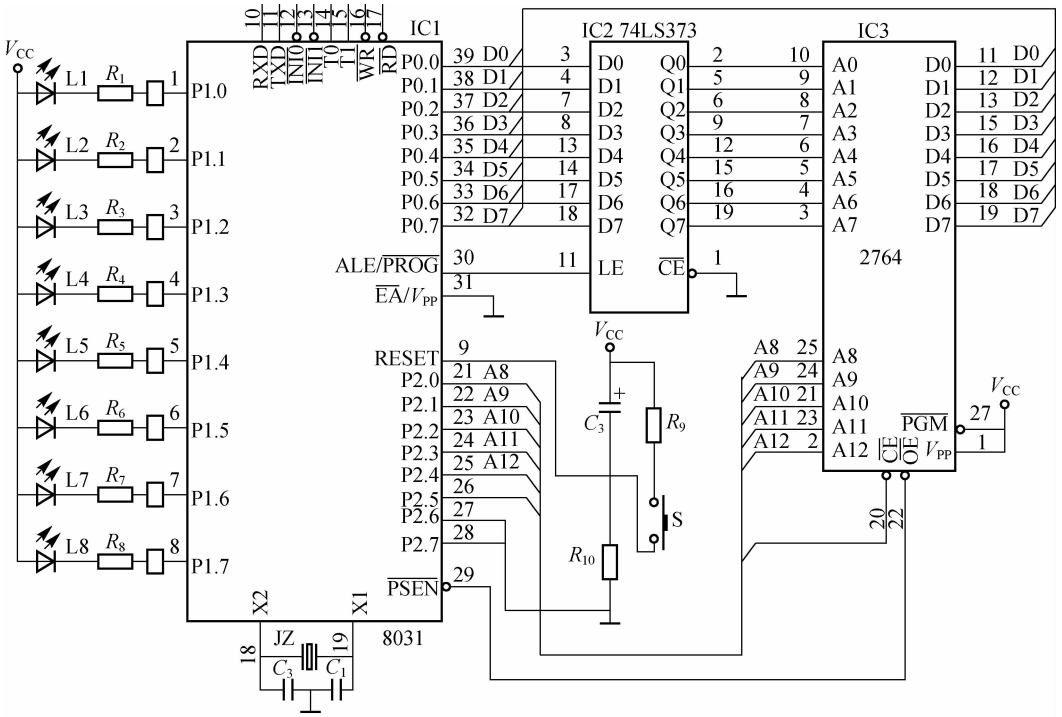


图3-6　彩灯控制器电路图

在实际应用中，还要根据以上原理图设计出 PCB 图，PCB 图用专用绘图软件 Protel99 做好后送到专门加工 PCB 的工厂加工出 PCB，然后到元器件市场购买所需要的元器件，再按原理图在 PCB 上焊接元器件，最后调试硬件。调试的方法是：先用万用表对照原理图每根线逐点检查，特别是电源不能接反，反复检查没错后，拔掉 8031 单片机，然后将 P1 口的每个引脚对地短路，看指示灯是否亮，如灯不亮，检查发光管是否损坏，方向是否接反，再查电阻是否开路，阻值是否太大，最后检查同相放大器是否损坏，如果有问题，一般换一个试试。检查没错后，一个实用的彩灯控制器的硬件设计成功。

对于这些小的实验设计可用 AT89C51 单片机或 AT89C2051 单片机，它们内部自带闪速存储器，可反复擦写，图 3-7 所示为用 AT89C51 单片机设计的电路图。从图中可见，省去了 74LS373 锁存器和 2764 芯片，使体积缩小、价格降低。需要注意的是，第 31 引脚通过一个电阻接高电平，复位电路、时钟电路不变。

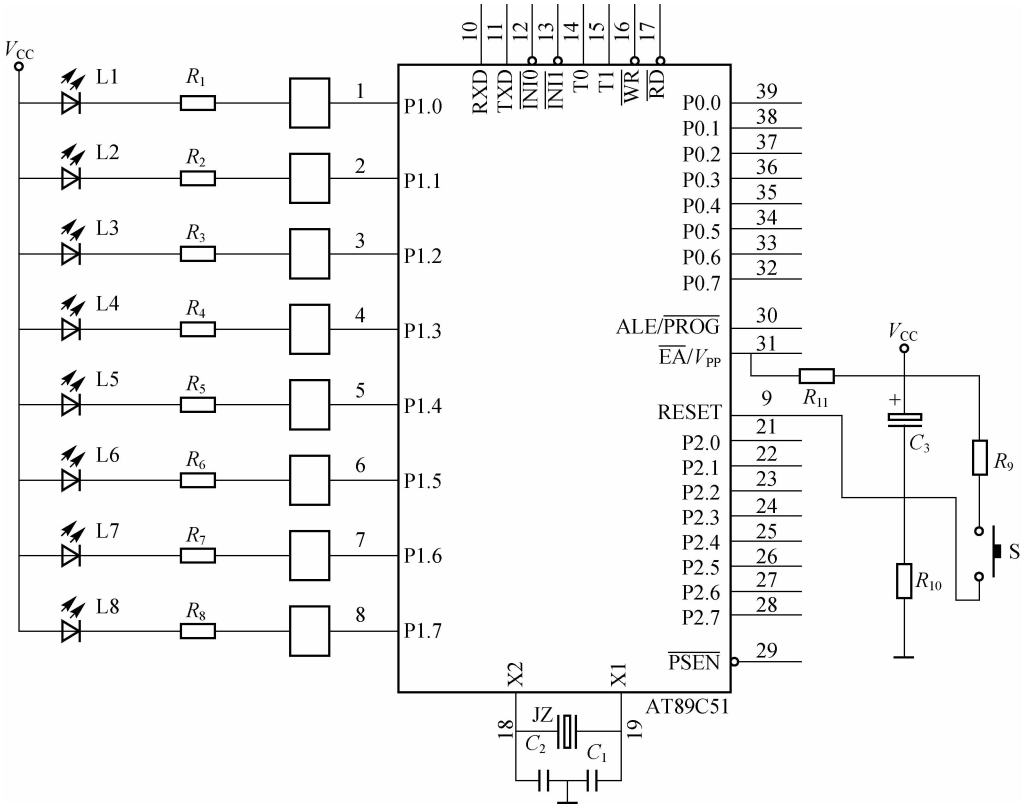


图 3-7　AT89C51 单片机设计彩灯控制器电路图

2）彩灯控制器软件设计

（1）延时程序。设计软件之前要考虑人眼的视觉暂留问题。指示灯一亮一暗的延时应在 0.5 s 以上，一般定 1 s，不然人的眼睛感觉不出亮暗变化。所以编程时两次向端口送输出数之间要延时，延时程序一般用循环程序编写（也可用定时器）。延时程序为

```
for(j=0;j<30000;j++);
```

(2)数据编码。从电路图可以知道,P1口为低电平时,指示灯亮;P1口为高电平时,指示灯不亮。在编程时用字节操作法,设定指示灯亮的对应引脚为低电平,指示灯不亮的对应引脚为高电平,把这些亮暗的情况先用十六进制编码,例如,要P1.0引脚控制的指示灯亮,其他的为暗,编码时暗的引脚为"1",亮的引脚为"0",即

P1.7	P1.6	P1.5	P1.4	P1.3	P1.2	P1.1	P1.0
1	1	1	1	1	1	1	0

以上的编码为二进制编码11111110B,一般将这种二进制编码转换为十六进制编码,转换方法常用8421编码,根据该编码,二进制编码11111110B转换为十六进制编码FEH,H表示此值为十六进制。

同样可编由P1.7引脚控制的指示灯亮,其他为暗时的码值

P1.7	P1.6	P1.5	P1.4	P1.3	P1.2	P1.1	P1.0
0	1	1	1	1	1	1	1

它的十六进制编码为7FH。

同样也可编两边亮,中间暗时的码值

P1.7	P1.6	P1.5	P1.4	P1.3	P1.2	P1.1	P1.0
0	1	1	1	1	1	1	0

它的十六进制编码为7EH。以此类推,可编各种情况的码值。例如,从上到下依次点亮的编码为FEH—FDH—FBH—F7H—EFH—DFH—BFH—7FH。从两端依次向中间点亮的编码为7EH—BDH—DBH—E7H。

(3)编程。掌握以上的基本知识后,可编写彩灯控制器的程序。例如,用字节操作的方法编写P1口控制的8个指示灯从上到下顺序点亮的程序如下。

```
#include <reg51.h>
#define uchar unsigned char
#define uint unsigned int
void Del()
{
    uint j;
    for(j=0;j<30000;j++);      //延时
}
void main()
{
    Lop:
        P1=0xfe;               //第1个指示灯亮
        Del();                 //调用延时子程序延时
        P1=0xfd;               //第2个指示灯亮
        Del();
```

```
        P1＝0xfb;                    //第 3 个指示灯亮
        Del();
        P1＝0xf7;                    //第 4 个指示灯亮
        Del();
        P1＝0xef;                    //第 5 个指示灯亮
        Del();
        P1＝0xdf;                    //第 6 个指示灯亮
        Del();
        P1＝0xbf;                    //第 7 个指示灯亮
        Del();
        P1＝0x7f;                    //第 8 个指示灯亮
        Del();
    goto Lop;
}
```

将此程序汇编后固化到 2764 芯片或 AT89C51 单片机中，彩灯从上到下依次点亮。还可以编写各种控制彩灯变化的程序，使彩灯的变化花样更多。

(4)调试程序。进入 Keil C 软件，建立项目，输入程序，对项目中的程序文件进行编译连接，并生成与项目文件同名的可执行代码及用于 EPROM 编程的 HEX 文件。μVision3 编译信息窗口如图 3-8 所示。0 个错误，0 个警告时，HEX 文件自动产生。只有 0 个错误时 HEX 文件才产生，有警告问题不大。若有错误，一一排除，直到完全没有错误为止。产生错误的主要原因有：在中文输入状态下输入；大小写没有区分；标点符号没有在英文输入状态下输入；语句后没有分号或多了符号。

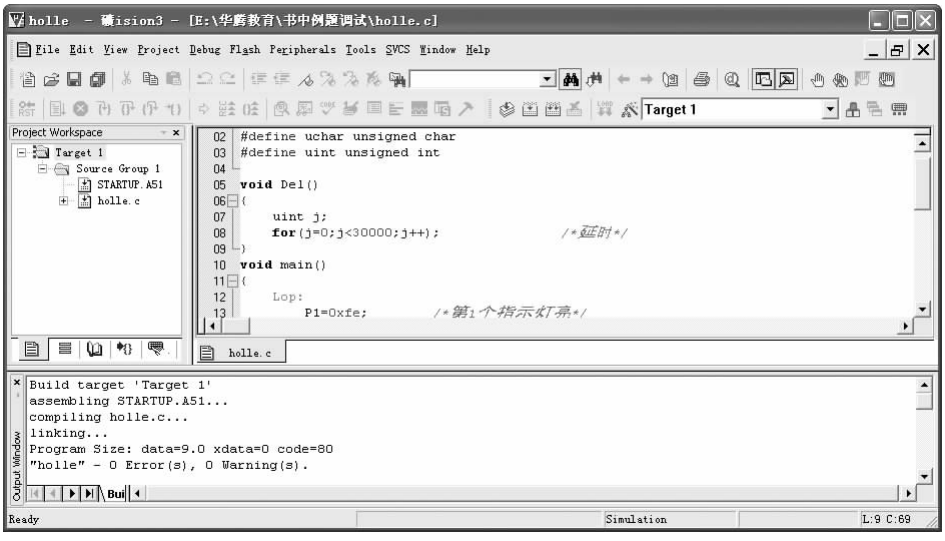


图 3-8　编译信息窗口

选择 Debug 菜单中的 Start/Stop Debug Session 选项，进入调试状态，如图 3-9 所示。

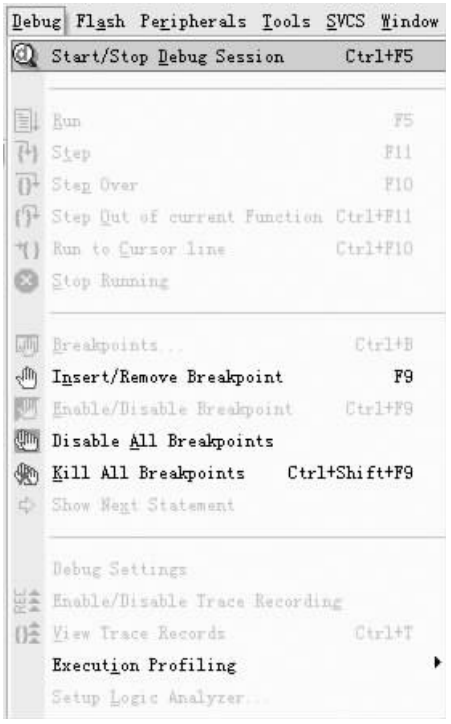


图 3-9　Debug 菜单中的 Start/Stop Debug session 选项

调试时，一般先用全速命令运行一次，看能否实现程序功能。若有错，要反复修改、调试程序。调试时，一般几个方法综合使用。单击 Debug/Start 按钮后，运行光标出现在程序开始位置，打开硬件 P1 口观察窗口，每按一下 F7 键，执行一条程序指令，直到第 8 个指示灯亮为止。将调试成功的程序汇编后固化到芯片中，若硬件不出问题，应能实现程序设计功能。

项目实践

1.8 只 LED 彩灯花样控制仿真

在 Proteus 软件中输入如图 3-10 所示的 8 只 LED 花样控制的原理图，然后装载之前调试成功的程序，全速运行程序，可观察到硬件联调结果：LED 从上到下依次点亮。还可编写 LED 从两端向中间依次点亮程序，从中间向两端依次点亮等花样控制程序。

2. LED 彩灯控制器硬件设计

1）PCB 制作

设计好 PCB 后要用覆铜板制作完成 PCB，制作 PCB 的方法一般有以下几种。

（1）PCB 制作厂家制作。用这种方法制作时，只要把制作的 PCB 设计图发到制作厂家，厂家就会按设计图做好，直接拿来焊接元器件即可。PCB 小批量制作价格较贵，而且制作时间较长，最低要一周时间，一般是集体提前制作。

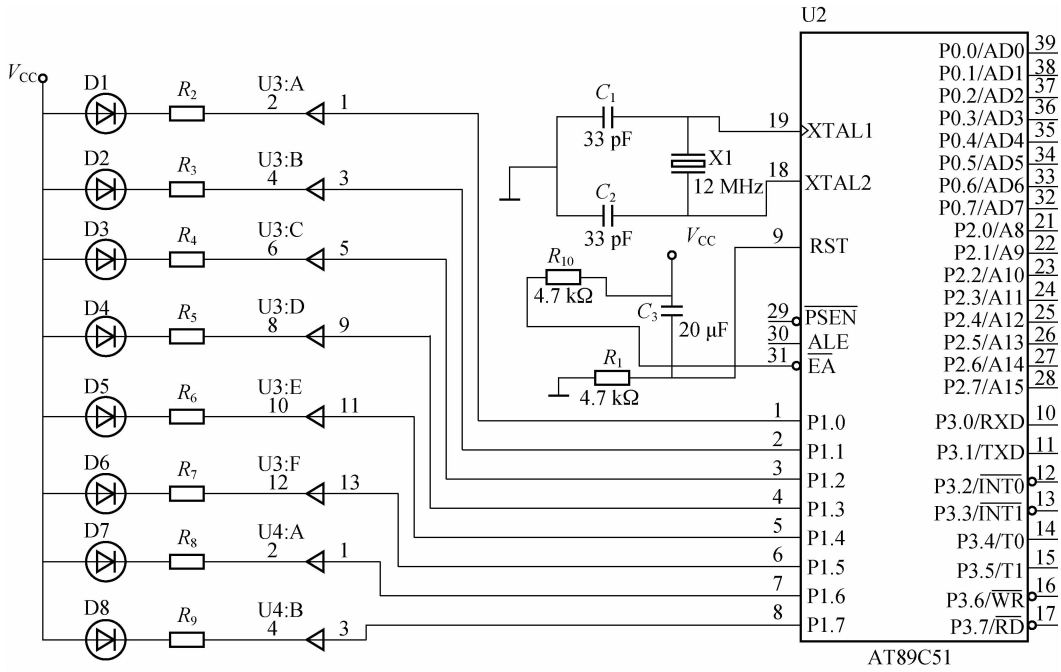


图 3-10　8 只 LED 灯花样控制原理图

(2)用雕刻机制作。若有雕刻机,首先要把 PCB 设计图转换成雕刻机所需的文件格式,并准备好雕刻头、覆铜板等工具和材料进行雕刻。雕刻好后打孔,作上锡处理,最后制作完成。

(3)自己手工制作。简单的 PCB 最好自己动手制作几次,这也是基本技能之一。先买好覆铜板(到电子元器件市场购买,最好买边角料板,能省钱)、三氯化铁、油漆和透明胶带等材料和工具,然后进行制作。制作方法是:对于单面板将透明胶带贴满覆铜板有铜的那面,再用小刀把要腐蚀的部分刻掉,保留不需腐蚀部分;将刻好的覆铜板放入三氯化铁中进行腐蚀,腐蚀完成后冲洗干净晾干;对腐蚀好的 PCB 打孔,做上锡处理,到此 PCB 制作完成。

(4)用万能板制作。最好买万能板自己连线焊接,这对于初学者来说特别重要。焊接时要用耐高温连接线连接。

本项目中的 PCB 要用万能板自己焊接。首先开列元器件清单,到市场买好元器件,然后按电路图焊接好。所有集成块均焊上插座,便于以后调试和检查。

2)电路调试

由于元器件特性参数的分散性,装配工艺的影响以及元器件缺陷和干扰等其他各种因素的影响,使得安装完毕的电子电路达不到设计要求的性能指标,这时需要通过调试来发现错误并纠正,使其达到预期的功能和技术指标。

调试的一般步骤为:初步调试,使电子电路处于正常工作状态;调整元器件的参数以及装配工艺分布参数,使电子电路处于最佳工作状态;在设计和元器件允许的条件下,改变内部因素、外部因素,如过压、过流、高温、连续长时间运行等,以检验电子电路的稳定性和可靠性,即所谓的考机。

调试的一般原则是先静态后动态。对于简单的电子电路,首先在电路未加输入信号的

直流状态下测试和调整各项技术性能指标，再输入适当的信号测试和调整各项技术性能指标。对于复杂的电子电路，需要先将复杂电路按各部分完成的功能分解成一些功能块（即单元电路），然后按照信号流程，按其功能原理进行功能检查，逐级进行调试。对具有精度要求的各项技术性能指标，必须用标准仪器仪表来检定，即所谓的分调。在分调的基础上再对电路整体的技术性能指标、波形参数进行测试调整，使之达到设计的要求，即所谓的总调。

拓展知识

PCB设计包括：确定其尺寸、形状、材料、外部连接和安装方法，确定布设导线和元器件的位置，确定印制导线的宽度、间距，确定焊盘的直径和孔径等。PCB设计必须符合电原理图的电气连接和电气、机械性能要求。PCB的面积大小应适中：过大时，印制线条长，阻抗增加，抗噪声能力降低，成本也高；过小时，则散热不好，并在线条间产生干扰。

1. 元器件布局的一般方法和要求

（1）元器件在PCB上的分布应尽量均匀，密度一致。无论是单面PCB还是双面PCB，所有元器件应尽可能都安装在同一面，以便加工、安装和维护。

（2）PCB上元器件的排列应整齐美观。一般应做到横平竖直，并力求元器件安装紧凑、密集，尽量缩短引线。如果装配工艺要求须将整个电路分割成几块安装时，应使每块装配好的PCB成为独立的功能电路，以便单独调试、检验和维护。

（3）元器件安装的位置应避免互相影响。元器件间不允许立体交叉和重叠排列，元器件的方向应与相邻印制导线交叉，电感元件要注意防电磁干扰，发热元件要放在有利于散热的位置，必要时可单独放置或装散热器，以降温和减少对邻近元器件的影响。

（4）变压器、扼流圈、大电容器、继电器等大而笨重的元器件可安装在主印制电路板之外的辅助底板上，利用附件将它们紧固，以利于加工和装配。也可将上述元器件安置在PCB靠近固定端的位置上，并降低重心，以提高机械强度，耐振、耐冲击力，减小PCB的负荷和变形。

（5）元器件的跨距（即元器件成型后两引线脚之间的距离）最大不应该超过元器件本体长度的2倍；单向引线的跨距，不应超过本体直径（或长度）的4/5，如图3-11所示。

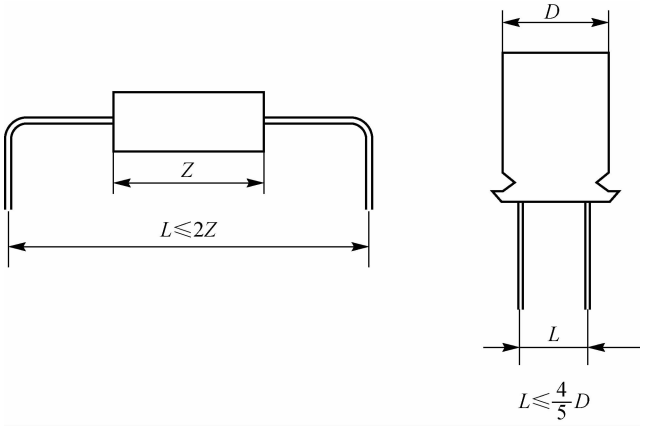


图3-11　元器件的跨距和单向引线的跨距

(6)元器件的间距。最小间距 d 等于相邻元件的半径(或厚度的一半)之和再加上安全间隙 b(b 为 1 mm/200 V),如图 3-12 所示。

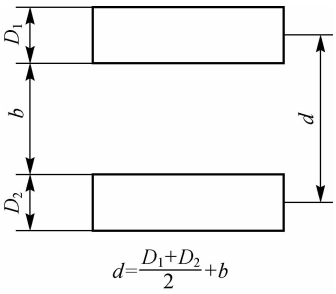


图 3-12　元器件的间距

2.布设导线的一般方法和要求

(1)公共地线应尽可能布置在 PCB 的最边缘,便于 PCB 安装以及与地相连。同时导线与 PCB 边缘应留有一定的距离,以便进行机械加工和提高绝缘性能。

(2)为减小导线间的寄生耦合,布线时应按信号的顺序进行排列,尽可能将输入线和输出线的位置远离,最好采用地线将两端隔开。输入线和电源线的距离应大于 1 mm,以减小寄生耦合。另外,输入电路的印制导线应尽量短,以减小感应现象及分布参数的影响。

(3)提供大信号的供电线和提供小信号的供电线应分开,特别是地线,最好是一点共地。

(4)高频电路中的高频导线,三极管各电极引线及信号输入、输出线应尽量做到短而直,易引起自激的导线应避免互相平行,宜采取垂直或斜交布线。若交叉的线较多,则最好采用双面板,将交叉的导线布设在 PCB 的两面。双面板的布线应避免 PCB 两面的印制导线平行,以减小寄生耦合。PCB 的两面导线最好成垂直或斜交布置。

3.印制导线的尺寸和图形

1)印制导线的宽度

同一块 PCB 上的印制导线宽度应尽可能保持均匀一致(地线除外)。印制导线的宽度主要与流过的电流大小有关,一般选择 1～2 mm,一些要流过大电流的电路,线宽要适当加大,可为 2～3 mm,公共地线和电源线在布线允许的情况下可为 4～5 mm。

2)印制导线的间距

印制导线的间距一般不小于 1 mm,当线间电压较高或通过高频信号时,其间距应相应增大,避免相对绝缘强度下降,分布电容增大。

3)印制导线的形状

印制导线的形状应简洁美观,如图 3-13 所示。在设计印制导线时应遵循以下三点。

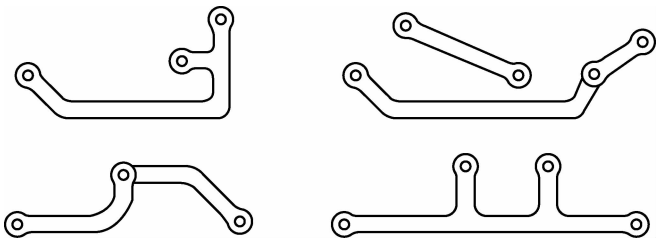


图 3-13　印制导线的形状

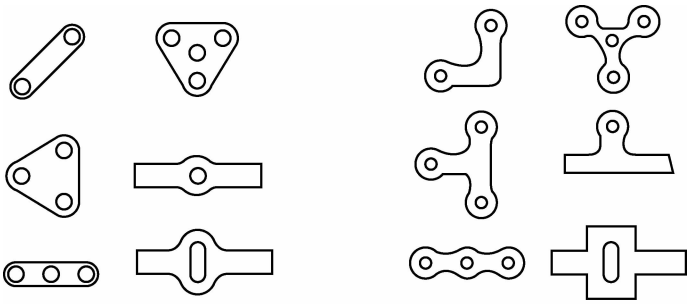
(1)除地线外,同一 PCB 上导线的宽度尽可能保持一致。

(2)印制导线的走向应平直,不应出现急剧的拐弯或尖角。

(3)应尽量避免印制导线出现分支。

4. 焊盘的形状和尺寸

为了增加在焊接元器件与机械加工时印制导线与基板的粘贴强度，必须将导线加工成圆形或岛形，如图 3-14 所示。环外径应略大于其相交的印制导线的宽度，通常取 2～3 mm。在单个焊盘或连接较短的两个接点时加一条辅助线，可增加接点的牢固。



(a)岛型　　(b)圆型

图 3-14　印制板接点的形状

复习思考题

1. 单片机中有哪些抗干扰技术？
2. LED 控制仿真时要注意哪些问题？