实验一　存储单元清零

一、实验目的

1.掌握存储器读写方法

2.了解存储器块的操作方法

二、实验说明

本实验指定某块存储器的起始地址和长度，要求能将其内容清零。通过该实验学生可以了解单片机读写存储器的方法，同时也可以了解单片机编程、调试方法。

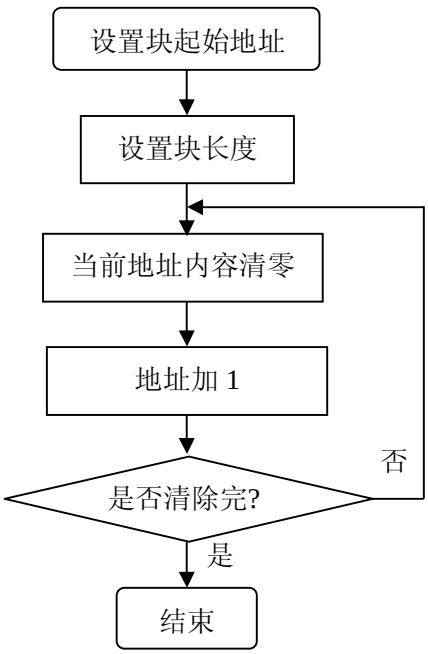
三、实验内容及步骤

1.启动 PC 机，打开 KEIL 软件（KEIL 软件使用附录使用说明书），软件设置为模拟调试状态。在所建的 Project 文件中添加 TH1.ASM 源程序进行编译，编译无误后，打开 View 菜单中的 Memory Window，在 Address 窗口输入 X:8000H 后回车，观察 8000H-80FFH，起始的 256 个字节单元的内容，可以发现这 256 个字节的内容都为 1。

2.打开 CPU 窗口，选择单步或跟踪执行方式运行程序，观察 CPU 窗口各寄存器的变化，可以看到程序执行的过程，加深对实验的了解，用户也可改变 A 的值，那输出的内容也会改变。

四、流程图及源程序

1.流程图



2.源程序

```
ORG     0000H
START   EQU     8000H
MOV     DPTR,   #START              ；起始地址
        MOV     R0，#0              ；设置 256 字节计数值
        MOV     A，#1H
Loop：
        MOVX    @DPTR，A
        INC     DPTR                ；指向下一个地址
        DJNZ    R0，Loop            ；计数值减一
        NOP
        LJMP    $
        END
```

五、思考题

若把 8000H-80FFH 地址单元内容改写为 56H，程序如何修改？

实验二　拆字实验

一、实验目的

掌握汇编语言的设计和调试方法

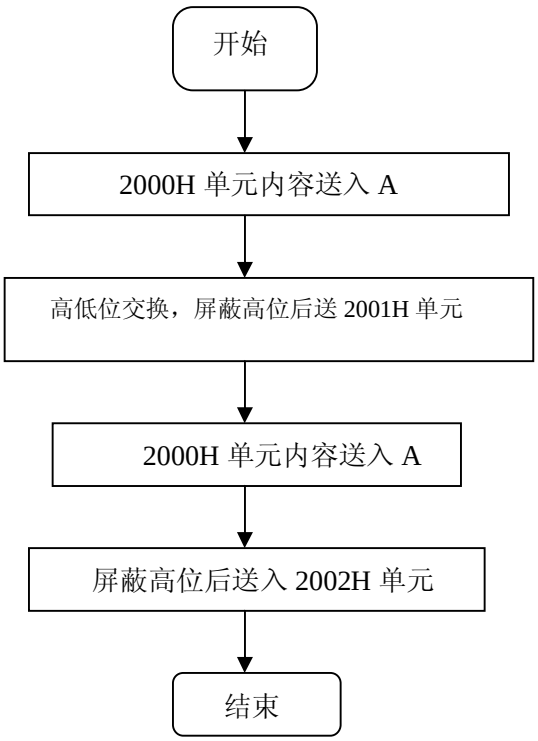
二、实验说明

本实验程序可看作计算机技术中的解压方法。我们把地址单元 2000H 中的内容拆开，其中高位内容送到地址单元 2001H 低位，低位内容送到地址单元 2002H 低位。地址单元 2001H、2002H 的高位清零。通常本程序用于将要显示的数据送到显示缓冲区时用。

三、实验内容及步骤

在地址单元 2000H 中任意输入一个参数，例如：56H，然后经过运算，检查地址单元 2001H、地址单元 2002H 是否分别为 05H、06H。如不是，说明程序有错，通过设置段点的方式进行调试，并检查最终结果。

四、流程图及编写源程序



五、思考题

如何灵活设置断点并通过设置段点的方式进行调试，并检查最终结果。

实验三　拼字实验

一、实验目的

进一步掌握汇编语言的设计和调试方法

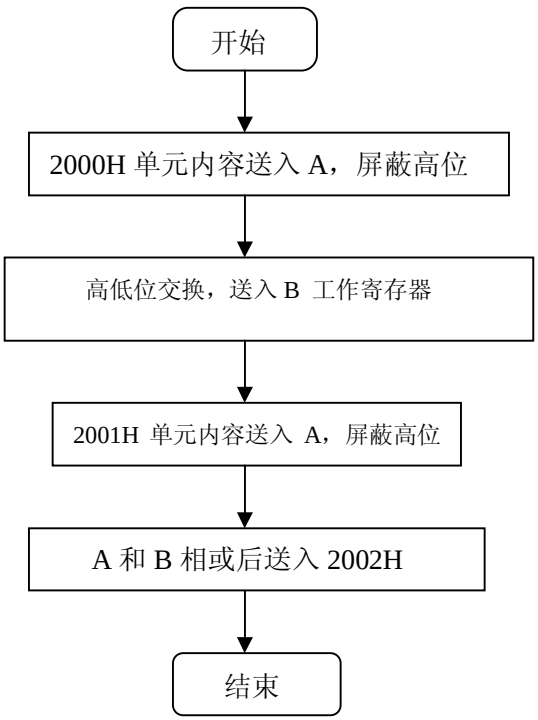
二、实验说明

本实验程序可看作计算机技术中的压缩方法。把地址单元 2000H、地址单元 2001H 的低位分别送人地址单元 2002H 高低位。本程序常用来把显示缓冲区的数据取出来拼装成一个字节。

三、实验内容及步骤

此实验程序的编写与上述实验正好相反，只是所用到的指令不尽相同。操作过程同上。

四、流程图及编写源程序



五、思考题

本实验如何用单步方式逐条检查运算结果？

实验四　数据传送实验

一、实验目的

掌握外部数据存储器 RAM 的数据传送

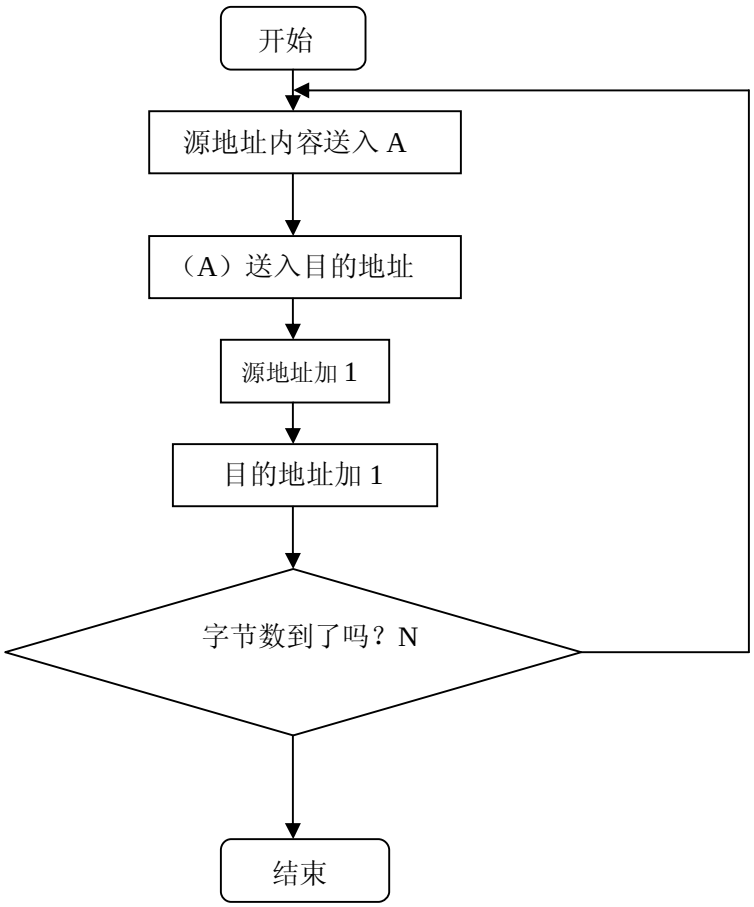
二、实验说明

把工作寄存器 R2、R3 的内容设置为外部**源数据**存储器 RAM 的首地址，把工作寄存器 R4、R5 的内容设置为外部**目的数据**存储器 RAM 的首地址。工作寄存器 R6、R7 的内容设置所要传送的字节数。最终检查两个不同存储空间的内容是否一致。

三、实验内容及步骤

工作寄存器 R2、R3 的内容设置为 1000H，工作寄存器 R4、R5 的内容设置为 4000H，工作寄存器 R6、R7 的内容设置为 1FFFH（1FFFH 为需要传送的字节数）。运行该程序后，当 1000H-2FFFH 地址空间的内容与 4000H-5FFFH 地址空间的内容一一对应时，程序正确。否则，通过设置段点的方式查错，修改错误的语句或逻辑。

四、流程图及编写源程序



五、思考题

工作寄存器 R6、R7 的内容起什么作用？1FFFH 化作 16 进制为多少？

实验五　程序跳转实验

一、实验目的

1.了解程序的多分支结构

2.掌握多分支结构程序的编程方法

二、实验说明

多分支结构是程序中常见的结构，在多分支结构的程序中，能够按调用号执行相应的功能，完成指定操作。若给出调用号来调用子程序，一般用查表方法，查到子程序的地址，转到相应子程序。

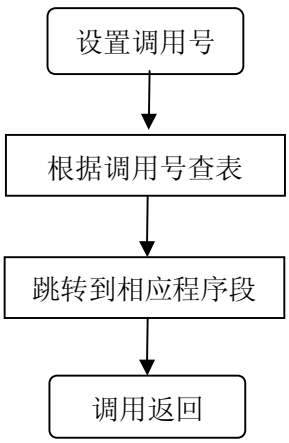
三、实验内容及步骤

1.启动 PC 机，打开 KEIL 软件，软件设置为模拟调试状态。在所建的 Project 文件中添加 TH4.ASM 源程序进行编译，编译无误后打开 View 菜单中的 Memory Window，在 Address 窗口输入 D:30H 后回车，点击运行按钮，观察地址 30H、31H、32H、33H 的数据变化，30H 更新为 0，31H 更新为 1，32H 更新为 2，33H 更新为 3。修改源程序中给 30H～33H 的赋值，重复实验，观察实验效果。

2.打开 CPU 窗口，选择单步或跟踪执行方式运行程序，观察 CPU 窗口各寄存器的变化，可以看到程序执行的过程，加深对实验的了解。

四、流程图及源程序（见光盘中的程序文件夹）

1.流程图



2.源程序

```
ORG     0000H
LJMP    START
FUNC0:  MOV   30H，#0
        RET
FUNC1:  MOV   31H，#1
```

```
        RET
FUNC2:  MOV  32H，#2
        RET
FUNC3:  MOV  33H，#3
        RET
FUNCENTER：
        ADD  A，ACC                ；AJMP 为二字节指令，调用号×2
        MOV  DPTR，#FUNCTAB
        JMP  @A+DPTR
FUNCTAB：
        AJMP  FUNC0
        AJMP  FUNC1
        AJMP  FUNC2
        AJMP  FUNC3
START：
        MOV   A，#0
        CALL  FUNCENTER
        MOV   A，#1
        CALL  FUNCENTER
        MOV   A，#2
        CALL  FUNCENTER
        MOV   A，#3
        CALL  FUNCENTER
        LJMP  $
        END
```

实验六　数据查找实验

一、实验目的

熟悉各类汇编语句及指令系统，掌握程序设计方法。

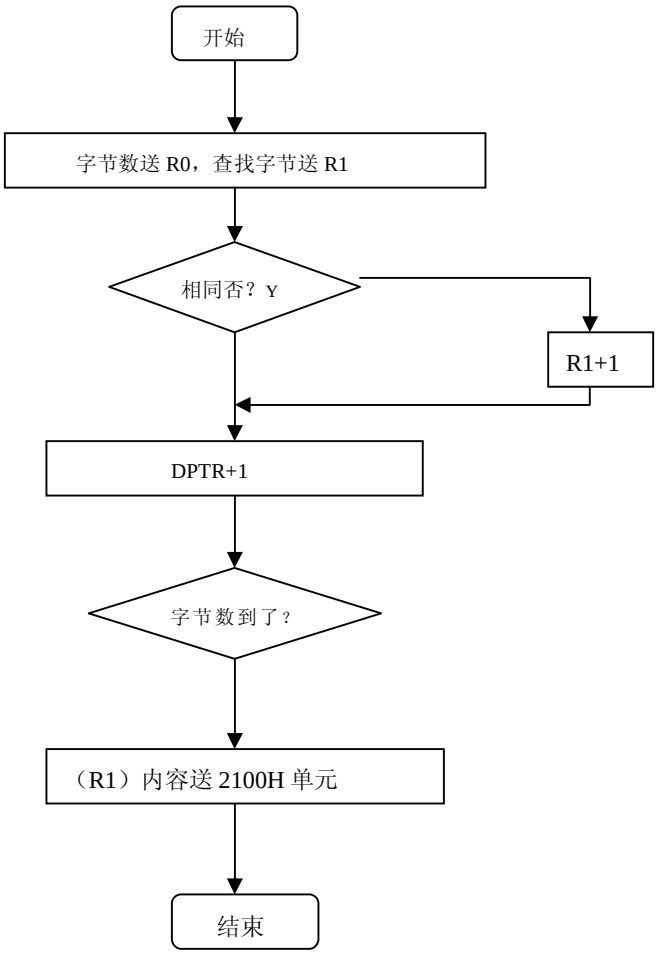
二、实验说明

在外部数据存储器 2000H-200FH 地址空间里存人 N 个零。通过编程，统计出为零的个数，并将结果存人外部数据存储器 2100H 单元。

三、实验内容及步骤

实验前，在外部数据存储器 2000H-200FH 地址空间里输人 N 个零。假定为 9 个零，其余不为零。运行本程序后检查外部数据存储器 2100H 单元为零的个数是 9 个吗？

四、流程图及编写源程序



五、思考题

单步运行与全速运行的区别？

实验七　I/O 口控制实验

一、实验目的

1、学习P1口的使用方法

2、学习延时子程序的编写和使用

二、实验说明

P1口是准双向口,它作为输出口时与一般的双向口使用方法相同。由准双向口结构可知当P1口用作输入口时，必须先对口的锁存器写“1”，若不先对它写“1”，读入的数据是不正确的。

三、实验内容及步骤

实验(一)：

用 P1 口做输出口，接八位逻辑电平显示，程序功能使发光二极管从右到左轮流循环点亮。

1、使用单片机最小应用系统。关闭该模块电源，用扁平数据线连接单片机 P1 口与八位逻辑电平显示模块 JD10。

2、用串行数据通信线连接计算机与仿真器，把仿真器插到模块的锁紧插座中，请注意仿真器的方向：缺口朝上。

3、打开 Keil uVision2 仿真软件，首先建立本实验的项目文件，接着添加“P1 口输出.ASM”源程序，进行编译，直到编译无误。

4、进行软件设置，选择硬件仿真，选择串行口，设置波特率为 38400。

5、打开模块电源和总电源，点击开始调试按钮，点击 RUN 按钮运行程序，观察发光二极管显示情况。发光二极管单只从右到左轮流循环点亮。

实验(二)：

用 P1.0、P1.1 作输入接两个拨断开关，P1.2、P1.3 作输出接两个发光二极管。程序读取开关状态，并在发光二极管上显示出来。

1、用导线分别连接单片机最小应用系统的 P1.0、P1.1 到两个拨断开关，P1.2、P1.3 到两个发光二极管。

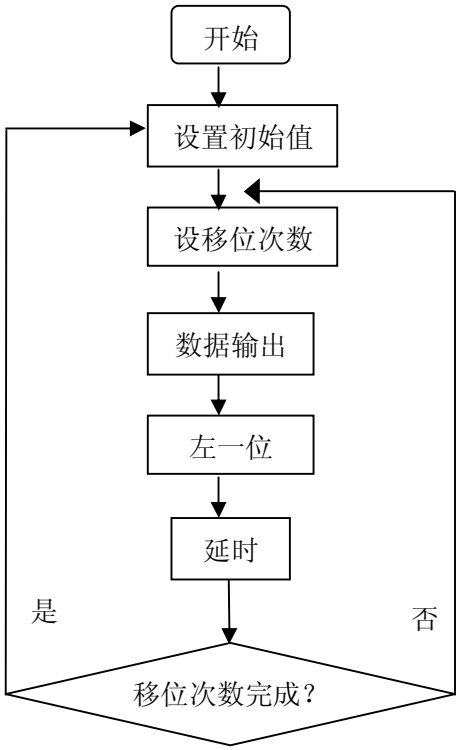
2、打开“P1_B.ASM”源程序，编译无误后，全速运行程序，拨动拨断开关，观察发光二极管的亮灭情况。向上拨为点亮，向下拨为熄灭。

3、也可以把源程序编译成可执行文件，把可执行文件用 ISP 烧录器烧录到 89S52/89S51 芯片中运行。（ISP 烧录器的使用查看附录二）

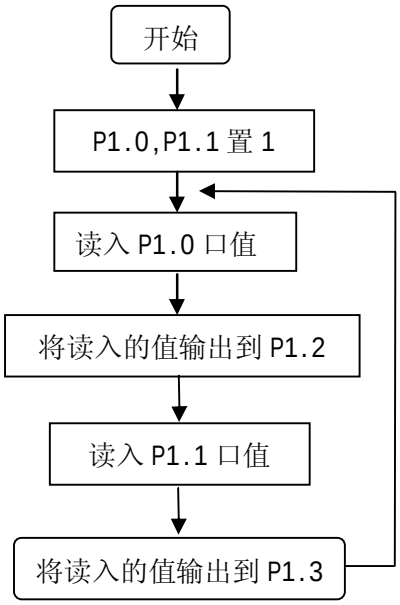
注：在做完实验时记得养成一个好习惯：把相应单元的短路帽和电源开关还原到原来的位置！以下将不在重诉。

四、流程图及编写源程序

1．流程图



(A)P1 口循环点灯程序框图



(B)P1 口输入输出程序框图

2．源程序

（一）实验一

（二）实验二

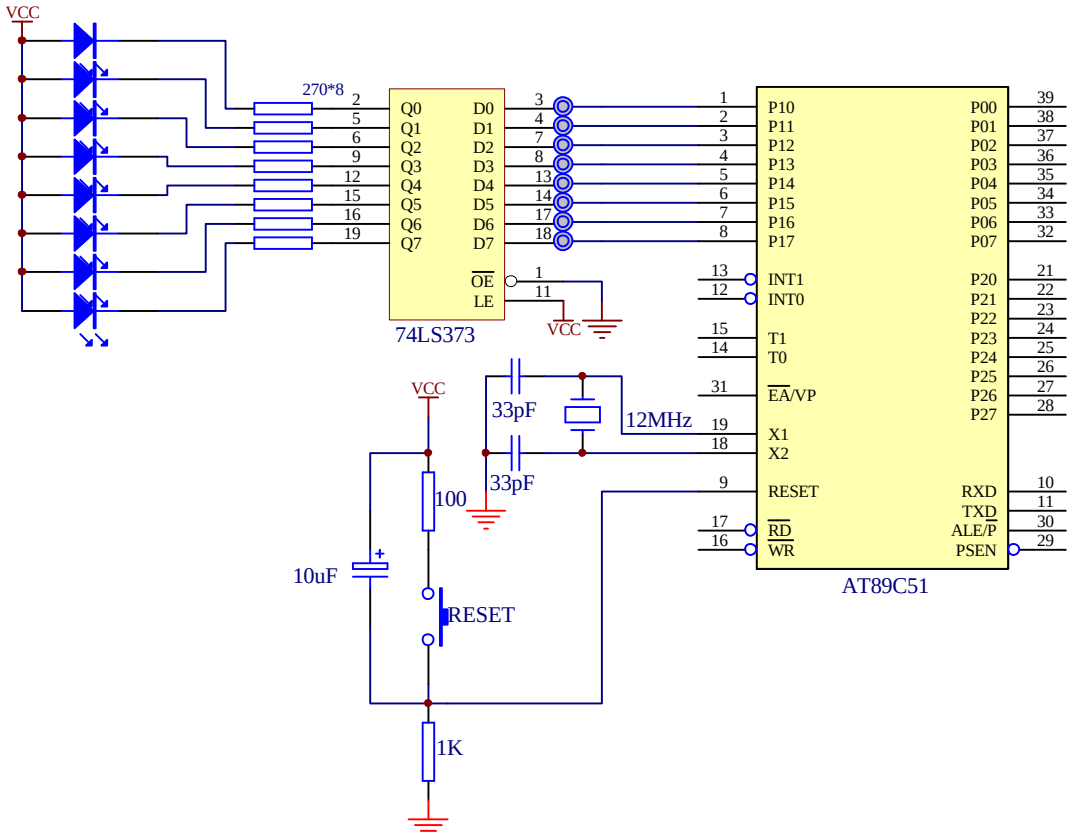
五、思考题

（1）对于本实验延时子程序

```
Delay：      MOV     R6，0FFH
             MOV     R7，0FFH
DelayLoop：DJNZ     R6，DelayLoop
             DJNZ    R7，DelayLoop
             RET
```

本模块使用 12MHz 晶振，粗略计算此程序的执行时间为多少？

六、电路图



实验八　外部脉冲宽度及频率测量实验

一、实验目的

1、 学习定时器/计数器的应用

2、 掌握外部脉冲宽度及频率的测量方法

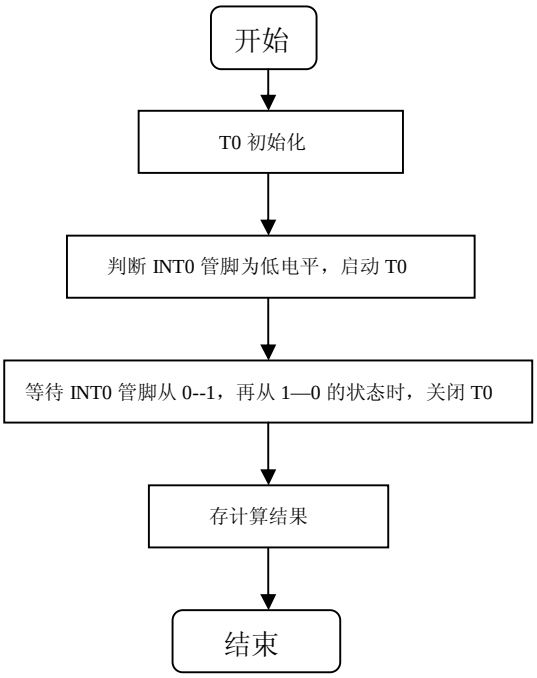
二、实验说明

关于内部定时/计数器的编程主要是定时常数的设置和有关控制寄存器的设置。内部计数器在单片机中主要有定时器和计数器两个功能。其相关的寄存器有寄存器 TMOD 和控制寄存器 TCON。TMOD 用于设置定时器/计数器的工作方式 0-3，并确定用于定时还是用于计数。TCON 主要功能是为定时器在溢出时设定标志位，并控制定时器的运行或停止等。

三、实验内容及步骤

把 F3 区的时钟信号 250K、125K、62.5K、31.25K 其中任一脉冲信号，接人单片机 CPU 的 P3.2 口（INT0），作为外部信号。关键的问题是要正确设置工作寄存器 TMOD 的参数，然后所编程序才能对外部信号进行宽度及频率的测量。程序执行完毕，读取 TH0、TL0 中的值。

四、流程图及源程序



五、思考题

对外部信号进行宽度及频率的测量是基于单片机的什么参数？

实验九　外部中断实验

一、实验目的

1、 掌握外部中断技术的基本使用方法

2、 掌握中断处理程序的编写方法

二、实验说明

外部中断的初始化设置有三项内容：中断总允许即 EA=1，外部中断允许即 EXi=1 (i=0,1)；中断方式设置有两种方式，即电平方式和脉冲方式。本实验选用后者，其前一次为高电平后一次为低电平时为有效中断请求；中断请求信号由引脚 INT0（P3.2）和 INT1（P3.1）引人 ，本实验由 INT0（P3.2）引人。**中断处理相关的控制寄存器有四个：**TCON、IE、SCON 及 IP。

三、实验内容及步骤

此实验涉及中断服务，需引人中断请求信号。为此将 G3 区域的负脉冲信号接人 CPU 的 INT0（P3.2）引脚，作为中断请求信号。当 S1G3 每按一次，即产生一个中断请求信号。为了证实中断响应与否，再将 C 区域 CPU 的 P1 口作为输出口，其中 P1.0、P1.1 引线到 E5 区域的电平显示引脚 L1、L2 上。程序要求，每中断响应一次，L1、L2 的灯交替闪烁一次，以此验证中断程序执行的状态。

四、画出流程图及编写源程序

五、思考题

本系统中断服务人口地址有几个？它们分别代表什么意义？

实验十　定时器输出 PWM 实验

一、实验目的

1、了解脉宽调制（PWM）的原理

2、学习用 PWM 输出模拟量

3、熟悉 51 系列单片机的延时程序

二、实验说明

PWM 是单片机上常用的模拟量输出方法，通过外接的转换电路，可以将脉冲的占空比变成电压。程序中通过调整占空比来调节输出模拟电压。占空比是制脉冲中高电平与低电平的宽度比。

三、实验内容及步骤

P1.0 输出 PWM 信号接转换电路，转换电压值送数字电压表显示。

1、选用 89C51 最小应用系统模块，用导线将 P1.0 接到 PWM 转换电压输入端，电压输出接电压表“+”端，电压表“-”端接地。

2、用串行数据通信线连接计算机与仿真器，把仿真器插到模块的锁紧插座中，请注意仿真器的方向：缺口朝上。

3、打开 Keil uVision2 仿真软件，首先建立本实验的项目文件，接着添加“PWM.ASM”源程序，进行编译，直到编译无误。

4、全速运行程序，观察电压表显示值，并做记录，程序默认是占空比 5:5 的 PWM。修改源程序 LOOP 程序段两次给累加器 A 的赋值，改为①“MOV　A，#1” ②“MOV　A，#9”，重新编译后运行，记录电压表显示值，这是占空比 1:9 的 PWM。同样，用户可做占空比 9:1 的 PWM，并做记录。比较三种 PWM 信号转换电压的大小，与理论值相比较。

5、也可以把源程序编译成可执行文件，把可执行文件用 ISP 烧录器烧录到 89S52/89S51 芯片中运行。（ISP 烧录器的使用查看附录二）

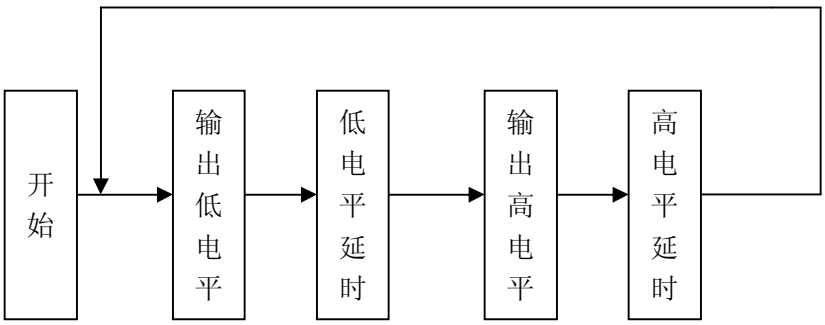
四、流程图及源程序

1.源程序清单：

```
        ；输出 50% (5:5) 占空比 PWM
        ；输出 10% (1:9) 占空比 PWM
        ；输出 90% (9:1) 占空比 PWM
        ORG   20H
OUTPUT  BIT   P1.0
LOOP:
        CLR   OUTPUT
        MOV   A, #5
        CALL  DELAY
        SERB  OUTPUT
        MOV   A, #5
        CALL  DELAY
        LJMP  LOOP
DELAY:
        MOV   R0, #0
```

```
DLOOP:
        DJNZ   R0, DLOOP
        DJNZ   ACC, DLOOP
        RET
        END
```

2.流程图

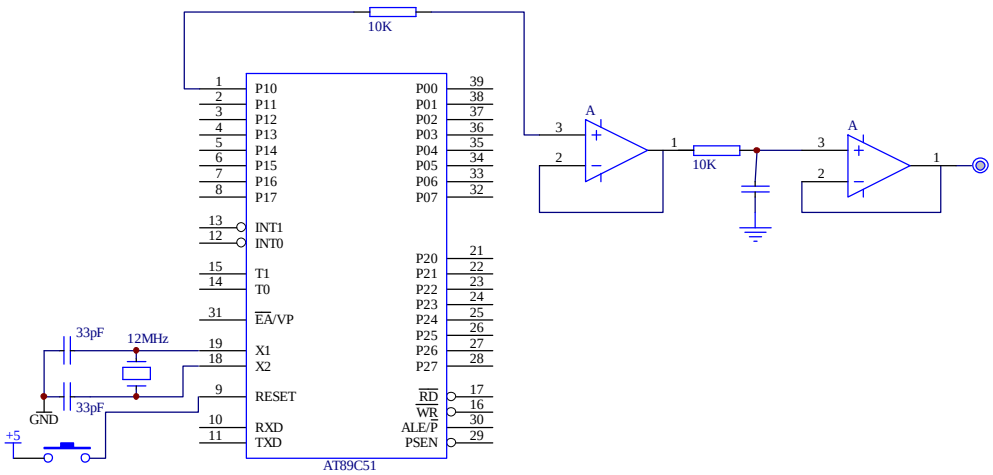


五、思考题

1、分析 PWM 转换电路的原理。

2、改变延时子程序 R0 的值，观察转换电压如何改变。

六、实验电路图



实验十一　8155 I/O 扩展实验

一、实验目的

1、了解 8155 芯片结构及接口方式

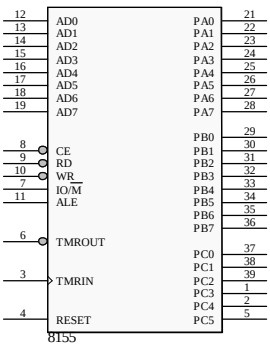
2、掌握 8155 输入、输出的编程方法

二、实验说明

1、本实验利用 8155 可编程并行口芯片，实现数据的输入、输出。实验中 8155 的 PA 口、PB 口作为输出口。与 8255 比，8155 具有更强的功能，因为它除能提供并行接口外还包括有 256 字节 RAM 存储器和 14 位定时器/计数器。8155 具有三个可编程 I/O 口，其中 PA、PB 为八位口，PC 口为 6 位口。PA 口、PB 口为通用的输入输出口，主要用于数据的 I/O 传送，他们都是数据口，因此只有输入输出两种工作方式。

2、了解实验用到的芯片引脚及功能：

8155 是一种可编程多功能接口芯片，功能丰富，使用方便，特别适合于扩展少量 RAM 和定时器/计数器的场合。其部分引脚功能如下：



8155 的引脚

(1) AD0～AD7——地址/数据总线，双向三态。

1)8155 有 256 字节静态 RAM,每一字节均有相应地址，输入输出数据均通过 AD0～AD7 口传送。

2）8155 内部有 6 个寄存器：A 口，B 口，C 口，命令状态寄存器，定时/计数器低 8 位，定时/计数器高 6 位加 2 位输出信号形式，6 个寄存器有各自相应的地址。地址及写入或读出的数据均通过 AD0～AD7 传送。

3）AD0～AD7 传送数据的方向由 RD,WR 信号控制。

(2) CE——片选信号，输入，低电平有效。

(3) WR——写信号，输入，低电平有效。

(4) RD——读信号，输入，低电平有效。

(5) PA0～PA7——A 口 8 位通用 I/O 线。

(6) PB0～PB7——B 口 8 位通用 I/O 线。

(7) PC0～PC5——C 口 6 位 I/O 线既可作通用 I/O 口，又可作 A 口和 B 口工作于选通方式下的控制信号。

(8) IO/M——I/O 与 RAM 选择信号。8155 内部 I/O 口与 RAM 是分开编址的，因此要使用控制信号进行区分。IO/M=0，对 RAM 进行读写；IO/M=1，对 I/O 进行和计数器进行读写。

3、本实验 8155 的端口地址由单片机的 P0 口和 P2.7 以及 P2.0 决定。控制口的地址为 7F00H；PA 口的地址为 7F01H；PB 口的地址为 7F02H。

三、实验内容及步骤

本实验分两种情况来进行：(一) PA 口作为输出口。 (二) PA 口作为输出口，PB 口作为输入口。

(一)PA 口作为输出口，接八位逻辑电平显示，程序功能使发光二极管单只从右到左轮流循环点亮。

1、单片机最小应用系统的 P0 口接 8155 的 D0～D7 口，8155 的 PA0～PA7 接八位逻辑电平

显示，单片机最小应用系统的 P2.0、P2.7、RD、WR、ALE 分别接 8155 的 IO/M、CE、RD、WR、ALE，RESET 接上最小系统的复位电路的 RESET。

2、用串行数据通信线连接计算机与仿真器，把仿真器插到模块的锁紧插座中，请注意仿真器的方向：缺口朝上。

3、打开 Keil uVision2 仿真软件，首先建立本实验的项目文件，接着添加 8155_A.ASM 源程序，进行编译，直到编译无误。

4、进行软件设置，选择硬件仿真，选择串行口，设置波特率为 38400。

5、也可以把源程序编译成可执行文件，把可执行文件用 ISP 烧录器烧录到 89S52/89S51 芯片中运行。（ISP 烧录器的使用查看附录二）

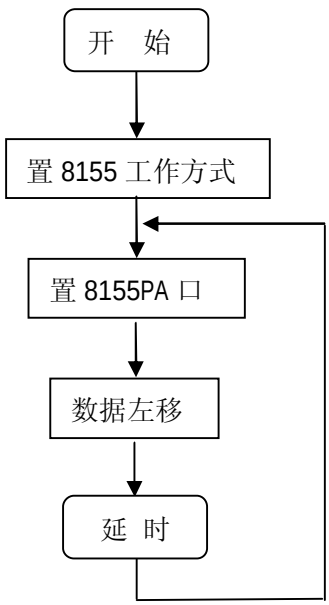
(二)PA 口作为输出口，PB 口作为输入口，PA 口读入键信号送八位逻辑电平显示模块显示。

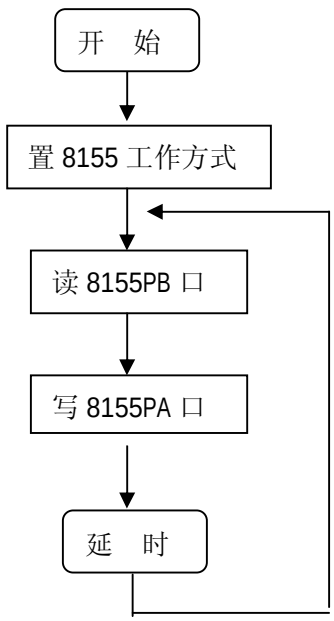
1、单片机最小应用系统的 P0 口接 8155 的 D0～D7 口，8155 的 PA0～PA7 接八位逻辑电平显示，PB0～PB7 口接查询式键盘模块，单片机最小应用系统的 P2.0、P2.7、$\overline{RD}$、$\overline{WR}$、ALE 分别接 8155 的 IO/M、CE、RD、WR、ALE，RESET 接上复位电路的 RESET。

2、打开 8155_B.ASM 源程序，编译无误后，全速运行程序。按查询式键盘各键，观察发光二极管的亮灭情况，发光二极管与按键相对应，按下为点亮，松开为熄灭。

5、也可以把源程序编译成可执行文件，把可执行文件用 ISP 烧录器烧录到 89S52/89S51 芯片中运行。（ISP 烧录器的使用查看附录二）

四、流程图及源程序





源程序如下：

（一）PA 口输出：

```
        ORG       0H
        PORTA     EQU     7F01H          ；A 口
        PORTB     EQU     7F02H          ；B 口
        CADDR     EQU     7F00H          ；控制字地址
        MOV       A,#03H                 ；方式 0，PA、PB 输出
        MOV       DPTR, #CADDR
         MOVX      @DPTR, A
LOOP：
        MOV       A, #0FEH
        MOV       R2, #8
OUTPUT：
        MOV       DPTR, #PORTA
        MOVX      @DPTR, A
        CALL      DELAY
        RL        A
        DJNZ      R2, OUTPUT
        LJMP      LOOP
DELAY:
        MOV       R6, #0




        MOV       R7, #0
DELAYLOOP：
        DJNZ      R6, DELAYLOOP
        DJNZ      R7, DELAYLOOP
        RET
        END
```

（二）PA 口输出，PB 口输入

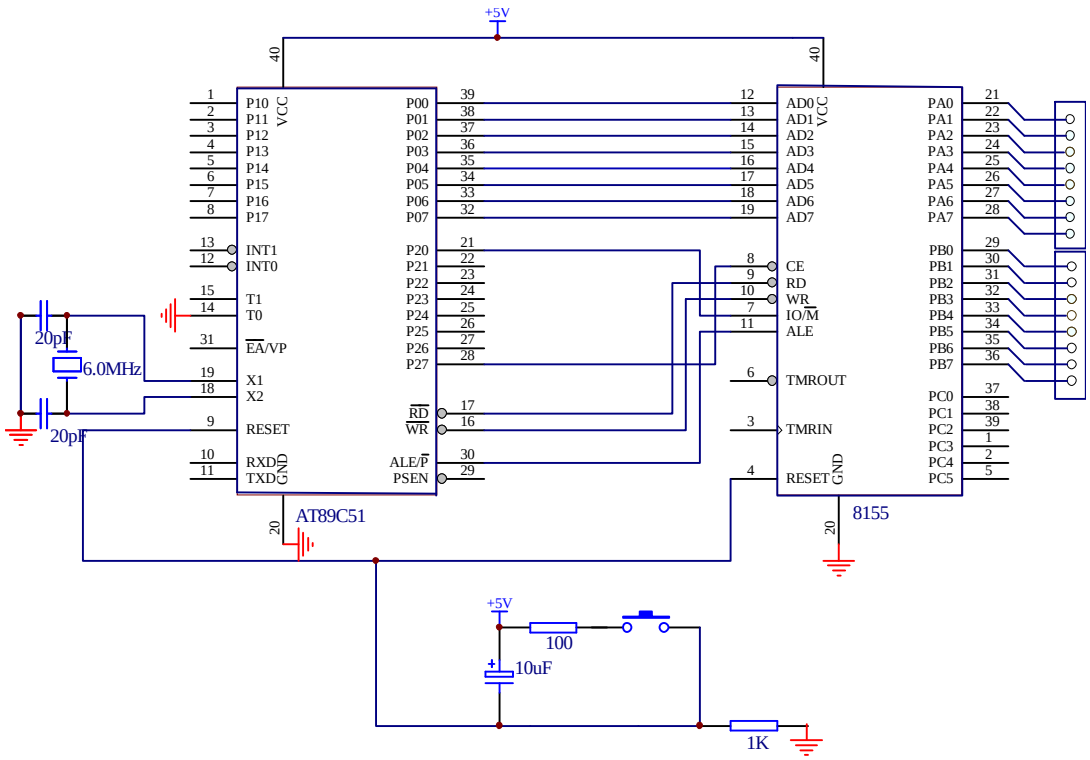
```
        ORG       0
        MODE      EQU    01H             ；方式 0，PA 输出，PB 输入
        PORTA     EQU    7F01H           ；A 口
        PORTB     EQU    7F02H           ；B 口
        CADDR     EQU    7F00H           ；控制字地址
        SJMP      START
         ORG       30H
        MOV       A, #MODE
        MOV       DPTR, #CADDR
```

```
        MOVX    @DPTR，A
START：MOV      DPTR，#PORTB
        MOVX    A，@DPTR            ；读入 B 口
        MOV     DPTR，#PORTA
        MOVX    @DPTR，A            ；输出到 A 口
        CALL    DELAY
        SJMP    START
        END
```

五、思考题

试用 8155PA 口作为输出口，PB 作为输入口，PC 作为输入口完成 8155 的输入、输出实验（其中 PA 口 LED 数码显示，PB 接拨断开关，PC 接查询式键盘实验模块）。

六、电路图



实验十二　继电器控制实验

一、实验目的

1、学习 I/O 端口的使用方法

2、掌握继电器的控制的基本方法

3、了解用弱电控制强电的方法

二、实验说明

现代自动控制设备中，都存在一个电子电路的互相连接问题，一方面要使电子电路的控制信号能控制电气电路的执行元件（电动机，电磁铁，电灯等），另一方面又要为电子线路和电气电路提供良好的电气隔离，以保护电子电路和人身的安全，继电器便能完成这一任务。

继电器电路中一般都要在继电器的线圈两头加一个二极管以吸收继电器线圈断电时产生的反电势。本电路的控制端为高电平时，继电器常开触点吸合，LED 灯被点亮当控制端口为低电平时，继电器不工作。

三、实验内容及步骤

用 P1.0 作为控制输出口，接继电器电路，使继电器重复吸合与断开。

1、使用单片机最小应用系统模块，用导线连接 P1.0 端口到继电器 CONTROL，K-OPEN 接八位逻辑显示的任意一个口，K-MID 接 GND。

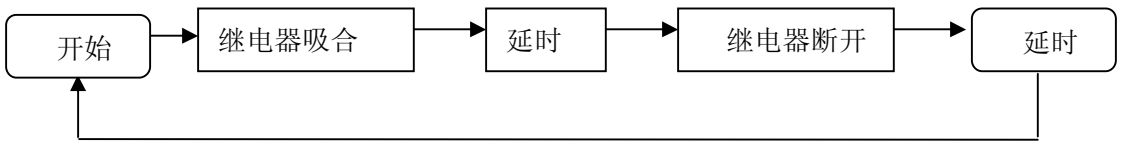
2、用串行数据通信线连接计算机与仿真器，把仿真器插到模块的锁紧插座中，请注意仿真器的方向：缺口朝上。

4、打开 Keil uVision2 仿真软件，首先建立本实验的项目文件，接着添加“继电器控制.ASM”源程序，进行编译，直到编译无误。

5、全速运行程序程序，观察二极管亮灭情况和听继电器开合的声音，继电器重复延时吸合与延时断开。

5、也可以把源程序编译成可执行文件，把可执行文件用 ISP 烧录器烧录到 89S52/89S51 芯片中运行。（ISP 烧录器的使用查看附录二）

四、流程图及编写源程序

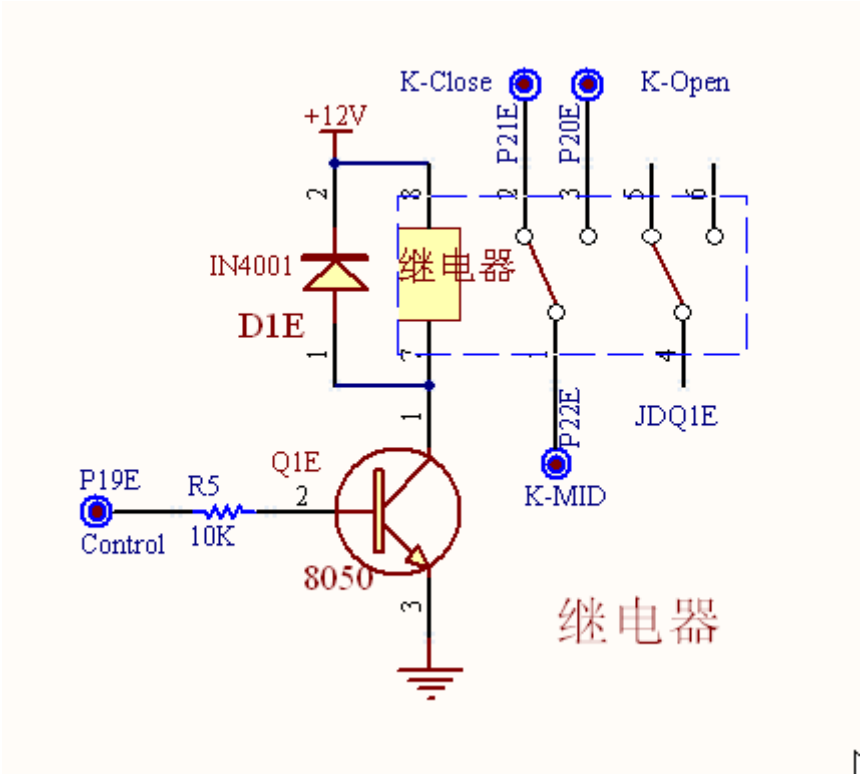


源程序清单：

五、思考题

试用单片机的其他输入输出口控制继电器。

六、电路图



实验十三　步进电动机控制实验

一、实验目的

1、掌握采用单片机控制步进电机的硬件接口技术。

2、掌握步进电机驱动程序的设计和调试方法。

3、熟悉步进电动机的工作特性。

二、实验说明

1、步进电动机有三线式、五线式、六线式三种，但其控制方式均相同，必须以脉冲电流来驱动。若每旋转一圈以 20 个励磁信号来计算，则每个励磁信号前进 18 度，其旋转角度与脉冲数成正比，正、反转可由脉冲顺序来控制。

2、步进电动机的励磁方式可分为全部励磁及半步励磁，其中全步励磁又有 1 相励磁及 2 相励磁之分，而半步励磁又称 1-2 相励磁。图为步进电动机的控制等效电路，适应控制 A、B、/A、/B 的励磁信号，即可控制步进电动机的转动。每输出一个脉冲信号，步进电动机只走一步。因此，依序不断送出脉冲信号，即可步进电动机连续转动。

a．1 相励磁法：在每一瞬间只有一个线圈导通。消耗电力小，精确度良好，但转矩小，振动较大，每送一励磁信号可走 18 度。若欲以 1 相励磁法控制步进电动机正转，其励磁顺序如图所示。若励磁信号反向传送，则步进电动机反转。

励磁顺序： A→B→C→D→A

STEP	A	B	C	D
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1

b．2 相励磁法：在每一瞬间会有二个线圈同时导通。因其转矩大，振动小，故为目前使用最多的励磁方式，每送一励磁信号可走 18 度。若以 2 相励磁法控制步进电动机正转，其励磁顺序如图所示。若励磁信号反向传送，则步进电动机反转。

励磁顺序： AB→BC→CD→DA→AB

STEP	A	B	C	D
1	1	1	0	0
2	0	1	1	0
3	0	0	1	1
4	1	0	0	1

c．1-2 相励磁法：为 1 相与 2 相轮流交替导通。因分辨率提高，且运转平滑，每送一励磁信号可走 9 度，故亦广泛被采用。若以 1 相励磁法控制步进电动机正转，其励磁顺序如图所示。若励磁信号反向传送，则步进电动机反转。

励磁顺序： A→AB→B→BC→C→CD→D→DA→A

STEP	A	B	C	D
1	1	0	0	0
2	1	1	0	0
3	0	0	1	0
4	0	1	1	0
5	0	0	1	0
6	0	0	1	1
7	0	0	0	1
8	1	0	0	1

3、电动机的负载转矩与速度成反比，速度愈快负载转矩愈小，当速度快至其极限时，步进电动机即不再运转。所以在每走一步后，程序必须延时一段时间。

三、实验内容及步骤

1、由单片机的 P1.0～P1.3 来控制步进电机,P2.0、P2.1、P2.2 控制电机的正转、反转、停止，最小系统的 P1 接步进电机的 P1 口，P2 口接查询式键盘的 JD3。

2、用串行数据通信线连接计算机与仿真器，把仿真器插到模块的锁紧插座中，请注意仿真器的方向：缺口朝上。

3、打开 KeiluVision2 仿真软件，首先建立本实验的项目文件，接着添加“步进电机控制

程序.asm”源程序，进行编译，直到编译无误。

4、编译无误后，运行程序，按下 KEY0 键电机正转,按下 KEY1 键电机反转,按下 KEY2 键电机停止。

5、也可以把源程序编译成可执行文件，用 ISP 烧录器烧到再烧录到 89S52/89S51 芯片中。（ISP 烧录器的使用查看附录二）

四、源程序

```
;//*************************************************************
;功能：  P1 口控制电机产生脉冲，P2.0，P2.1，P2.3 口接查询失件盘控制电机的
;        反转，正转，停止控制。

;接线：P1 口接步进电机的 P1 口，最小系统的 P2 口接查询式键盘的 JD13。

;//*************************************************************
        ORG   0000
STOP:   ORL   P1,  #00H     ; 步进电机停止
LOOP:   MOV   P1,  #00H
        JNB   P2.0,FOR2     ; 如果 p2.0 按下正转
        JNB   P2.1,REV2     ; 如果 p2.1 按下反转
        JNB   P2.2,STOP1    ; 如果 p2.2 按下停止
        JMP   LOOP          ;反复监测键盘
FOR:    MOV   R0,  #00H     ;正转到 tab 取码指针初值
for1:   MOV   A,R0          ;取码
        MOV   DPTR,#TABLE   ;
        MOVC  A,@A+DPTR
        JZ    FOR           ;是否到了结束码 00h
        CPL   A             ;把 acc 反向
        MOV   P1,A          ;输出到 p1 开始正转
        JNB   P2.2,STOP1    ;如果 p2.2 按下停止
        JNB   P2.1,REV2     ;如果 p2.1 按下正转
        JNB   P2.0,FOR2     ;如果 p2.0 按下反转
        CALL  DELAY         ;转动的速度
        INC   R0            ;取下一个码
        JMP   FOR1          ;继续正转
rev:    MOV   R0,#05H       ;反转到 tab 取码指针初值
rev1:   MOV   A,R0
        MOV   DPTR,#TABLE   ;取码
        MOVC  A,@A+DPTR
        JZ    REV           ;是否到了结束码 00h
        CPL   A             ;把 acc 反向
```

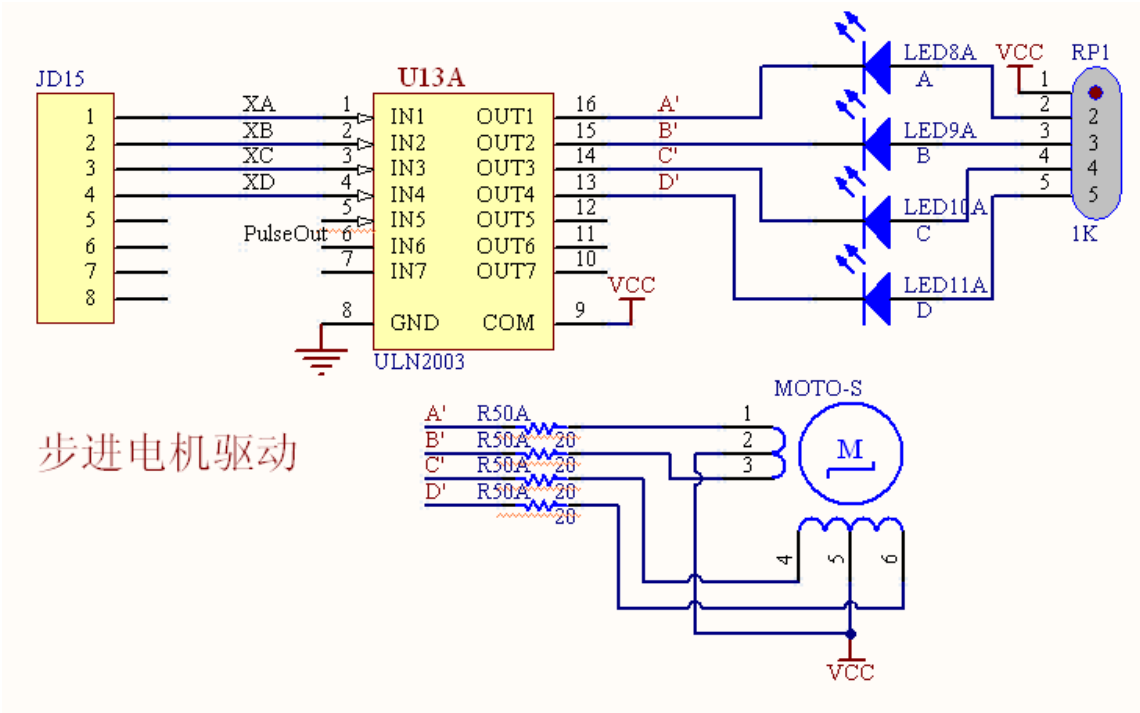
```
        MOV   P1,A          ;输出到 p1 开始反转
        JNB   P2.2,STOP1    ;如果 p2.2 按下停止
        JNB   P2.1,REV2     ;如果 p2.1 按下正转
        JNB   P2.0,FOR2     ;如果 p2.0 按下反转
        CALL  DELAY         ;转动的速度
        INC   R0            ;取下一个码
        JMP   REV1          ;继续反转
stop1:  CALL  DELAY         ;按 p3.4 的消除抖动
        JNB   P2.2,$        ;p2.2 放开否?
        MOV   P1,  #00H
        CALL  DELAY         ;放开消除抖动
        JMP   STOP
for2:   CALL  DELAY         ;按 p2.0 的消除抖动
        JNB   P2.0,$        ;p2.0 放开否?
        CALL  DELAY         ;放开消除抖动
        JMP   FOR
rev2:   CALL  DELAY         ;按 p2.1 的消除抖动
        JNB   P2.1,$        ;p3.3 放开否?
        CALL  DELAY         ;放开消除抖动
        JMP   REV

DELAY:  MOV   R1,#900       ;步进电机的转速 20ms
D1:     MOV   R2,#900
        DJNZ  R2,$
        DJNZ  R1,D1
        RET
TABLE:
        db 01h,02h,04h,08h  ;正转表
        db 00               ;正转结束
        db 01h,08h,04h,02h  ;反转
        db 00               ;反转结束
        END
```

五、思考问题

如何改变电机的工作方式或者方向、速度，设计控制软件？

六、实验电路图



实验十四　并行 A/D 转换实验

一、实验目的

1．掌握 ADC0809 模/数转换芯片与单片机的连接方法及 ADC0809 的典型应用；

2．掌握用查询方式、中断方式完成模/数转换程序的编写方法。

二、实验内容

利用系统提供的 ADC0809 接口电路，实现单片机模数转换。模拟信号为 0～5V 电位器分压输出，单片机控制 ADC0809 读取模拟信号，并在数码管上用十进制形式显示出来。

三、实验要求

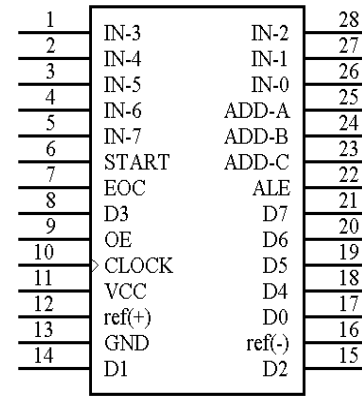
根据实验内容编写一个程序，并在实验仪上调试和验证。

四、实验说明和电路原理图

1）A/D 转换芯片 ADC0809 简介

本实验使用 ADC0809 模数转换器，ADC0809 是 8 通道 8 位 CMOS 逐次逼近式 A/D 转换芯片，片内有模拟量通道选择开关及相应的通道锁存、译码电路，A/D 转换后的数据由三态锁存器输出，由于片内没有时钟需外接时钟信号。

芯片的引脚如右图，各引脚功能如下：



IN0～IN7：八路模拟信号输入端。

ADD-A、ADD-B、ADD-C：三位地址码输入端。

CLOCK：外部时钟输入端。CLOCK 输入频率范围在 10～1280KHz，典型值为 640KHz，此时 A/D 转换时间为 100us。51 单片机 ALE 直接或分频后可与 CLOCK 相连。

D0～D7：数字量输出端。

OE：A/D 转换结果输出允许控制端。

当 OE 为高电平时，允许 A/D 转换结果从 D0～D7 端输出。ADC0809 引脚图

ALE：地址锁存允许信号输入端。

八路模拟通道地址由 A、B、C 输入，在 ALE 信号有效时将该八路地址锁存。

START：启动 A/D 转换信号输入端。

当 START 端输入一个正脉冲时，将进行 A/D 转换。

EOC：A/D 转换结束信号输出端。

当 A/D 转换结束后，EOC 输出高电平。

Vref(+)、Vref(-)：正负基准电压输入端。

基准正电压的典型值为+5V。

2）本实验需要用到 80C51 MCU 模块（E 区）、静态数码显示模块（A4 区）、可调电源模块（D2 区）和 A/D 转换模块（B7 区）。A/D 转换电路原理参考图 22.1

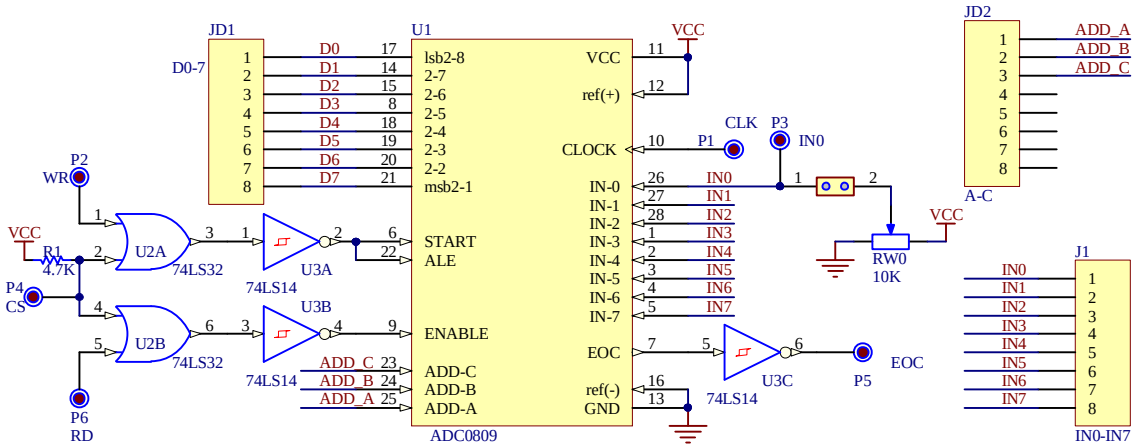


图 23.1 ADC0809 模数转换电路

五、实验预习要求

学习教材的相关内容，根据实验要求画出程序流程图，写出实验程序。

六、实验步骤

1）用 8P 数据线分别连接 80C51 MCU 模块的 JD0（P0 口）、JD4（A0～7 口）到 A/D 转换模块的 JD1B7（D0～7 口）、JD2B7（A0～7 口）；用导线分别连接 80C51 MCU 模块的 WR、RD、P2.0、ALE、INT1 到 A/D 转换模块的 WR、RD、CS、CLK、EOC；AIN0 接可调电源模块的 0－5V 输出端（AIN0 也可在程序运行之后接）；80C51 MCU 模块的 P10、P11 分别接静态数码显示模块的 DIN、CLK。

2）用 USB 数据线连接计算机与实验箱的仿真接口（CON2）；**将直流稳压电源模块的直流控制开关 S1 打到 ON，将 80C51 MCU 模块的电源钮子开关 S2 拨到上端；将 A/D 转换模块的 JPB7 置位到上端。**

3）启动 PC 机，打开 Keil 仿真软件，建立本实验的项目文件，输入源程序，用仿真器调试程序。

4）实验现象：静态数码显示模块显示“Ad XX”，“XX”为 Ad 转换后的值，调节可调电源模块的电位器 RW1D2，显示值随着变化，顺时针旋转值增大，Ad 转换值的范围是 00～FF。

5）把 Keil 仿真软件生成的可执行文件（hex 文件）用 ISP 下载器烧录到 AT89S52 芯片中运行，JP3 跳线器置位上方（EA 为高电平）。

七、实验参考程序

```
                dbuf0   equ     30h
                temp    equ     40h
                org     0000h
                ljmp    start
                org     0100h
start:          mov     35h,#11h
                mov     36h,#11h
                mov     37h,#11h
                mov     r0,#dbuf0
                mov     @r0,#0ah
                inc     r0
                mov     @r0,#0dh
                inc     r0
                mov     @r0,#11h
                inc     r0
                mov     dptr,#0fef0h
                mov     a,#0
                movx    @dptr,a
wait:           jb      p3.3,wait
                movx    a,@dptr
                mov     p1,a
                mov     b,a
                swap    a
                anl     a,#0fh
                xch     a,@r0
                inc     r0
                mov     a,b
                anl     a,#0fh
```

```
                xch     a,@r0
                acall   disp1
                acall   delay
                ajmp    start
disp1:          mov     r0,#dbuf0
                mov     r1,#temp
                mov     r2,#8
dp10:           mov     dptr,#segtab
                mov     a,@r0
                movc    a,@a+dptr
                mov     @r1,a
                inc     r0
                inc     r1
                djnz    r2,dp10
                mov     r0,#temp
                mov     r1,#5
dp12:           mov     r2,#8
                mov     a,@r0
dp13:           rlca
                mov     0b0h,c
                clr     0b1h
                setb    0b1h
                djnz    r2,dp13
                inc     r0
                djnz    r1,dp12
                ret
segtab:         db3fh,06h,5bh,4fh,66h,6dh
                db7dh,07h,7fh,6fh,77h,7ch
                db58h,5eh,79h,00h,00h
delay:          mov     r4,#0ffh
aa1:            mov     r5,#0ffh
aa:             nop
                nop
                djnz    r5,aa
                djnz    r4,aa1
                ret
                end
```

附录　THKL-C51 仿真器联机及软件的使用说明

一、仿真器自检步骤

不要带电插拔串口，以防止由此产生的浪涌电流损坏 MAX232 通讯芯片，下面的操作顺序可以避免带电插拔。

联机正确顺序：插好仿真用串口旋紧固定螺栓>>插上 USB 电源接口>>连接目标硬件，可以是任何 51 系统开发板、试验板、工控板、目标板...等等的 51 硬件系统。

脱机正确顺序：拔下 USB 电源接口>>拔下仿真用串口。如果短期内经常要使用仿真功能，无需拔下串口。

因为仿真器在通电瞬间要对系统进行自检，所以在通过 USB 给仿真系统供电之前，仿真头上不要连有负载。接通 USB 电源，自检通过后 POW LED 指示灯会亮起来，表示自检通过，此时就可以进入的硬件仿真了。

二、仿真器复位按钮的作用

在仿真器的右侧下方有一个小的按纽，这个按钮用来给整个仿真器硬件系统复位，什么时候需要按这个按钮呢？设置好 KEIL 的硬件环境后，在每次点击  进入仿真环境之前，需要按一下这个复位按纽，这样 KEIL 启动后，软件和已复位的硬件仿真器就会顺利联机，在点击  进入仿真环境之后，仿真器完全由 KEIL 控制，此时不要按这个按钮，否则在仿真过程中系统将会提示联机中断。

如果需要给硬件复位的话，请先点击仿真器的复位键然后点  退出 KEIL 仿真调试环境。

仿真器使用注意事项：在打开 PC 机之前请把仿真器和 PC 机的串口连好。在联机后，请千万不要带电插拔仿真器和 PC 机的接口，如果带电插拔仿真器就可能导致接口电路 MAX232 损坏。注意插拔的时候仿真器或者 PC 机至少有一方的电源是断开的。PC 机的串口和并口等接口的最大不便就是不支持热插拔，这也是开发 USB 接口的根本原因。

断开连接之前推荐步骤：

1.按一下仿真器硬件复位按纽

2.按  退出仿真环境

3.关闭 KEIL,关闭 PC 机，最后再断开硬件连接,如果要经常使用则不用断开硬件连接。

三、Keil uVision2 仿真软件的使用说明

μVision2 集成开发环境

μVision2 IDE 是德国 Keil 公司开发的基于 Windows 平台的单片机集成开发环境，它包含一个高效的编译器、一个项目管理器和一个 MAKE 工具。其中 Keil C51 是一种专门为单片机设计的高效率 C 语言编译器，符合 ANSI 标准，生成的程序代码运行速度极高，所需要的存储器空间极小，完全可以与汇编语言媲美。

1．关于开发环境

μVision2 的界面如图 1－1 所示，μVision2 允许同时打开、浏览多个源文件。

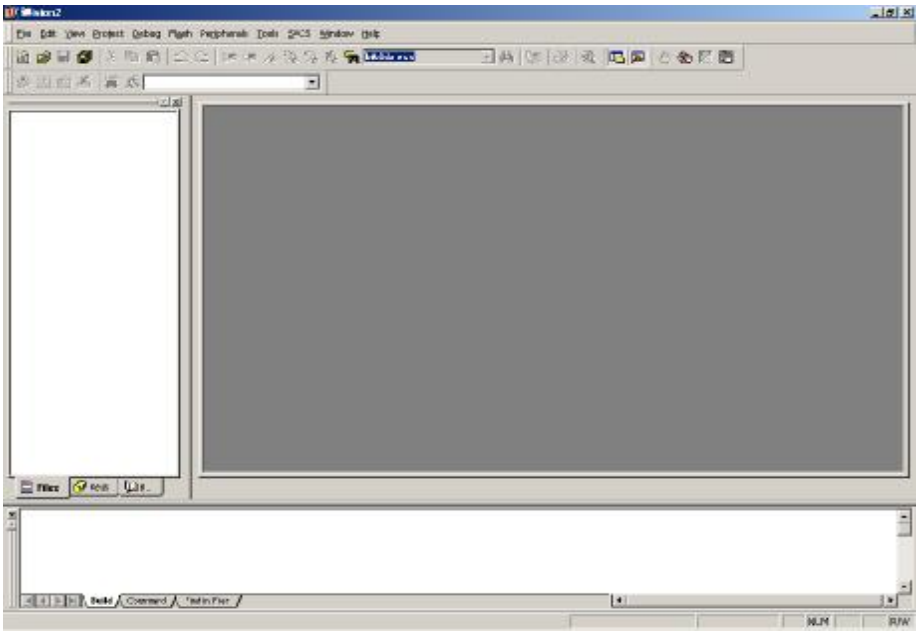


图 1—1 μVision2 界面图

2．菜单条、工具栏和快捷键

下面的表格列出了 μVision2 菜单项命令，工具栏图标，默认的快捷以及他们的描述。

1）编辑菜单和编辑器命令 Edit（如表 1-1 所示）

表 1-1　　编辑菜单和编辑器命令 Edit

菜单	工具栏	快捷键	描述
Home			移动光标到本行的开始
End			移动光标到本行的末尾
Ctrl+Home			移动光标到文件的开始
Ctrl+End			移动光标到文件的结束
Ctrl+<-			移动光标到词的左边
Ctrl+->			移动光标到词的右边
Ctrl+A			选择当前文件的所有文本内容
Undo		Ctrl+Z	取消上次操作
Redo		Ctrl+Shift+Z	重复上次操作
Cut		Ctrl+X Ctrl+Y	剪切所选文本 剪切当前行的所有文本
Copy		Ctrl+C	复制所选文本
Paste		Ctrl+V	粘贴
Indent Selected Text			将所选文本右移一个制表键的距离
Unindent Selected Text			将所选文本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
GotoPrevious 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			在当前文件中查找文本
		F3	向前重复查找
		Shift+F3	向后重复查找
		Ctrl+F3	查找光标处的单词
		Ctrl+]	寻找匹配的大括号、圆括号、方括号（用此命令将光标放到大括号、圆括号或方括号的前面）
Replace			替换特定的字符
Find in Files…			在多个文件中查找
Goto Matching brace			选择匹配的一对大括号、圆括号或方括号中的内容

2）选择文本命令

在 μVision2 中，可以通过按住 Shift 键和相应的键盘上的方向键来选择文本。如 Ctrl+->可以移动光标到下一个词，那么，Ctrl+Shift+->就是选择当前光标位置到下一个词的开始位置间的文本。当然，也可以用鼠标来选择文本。

3）项目菜单 Project 和项目命令 Project（如表 1-2 所示）

表 1-2　　项目菜单和项目命令 Project

菜单	工具栏	快捷键	描述
New Project…			创建新项目
Import μ Vision1 Project…			转化 μ Vision1 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			维护一个项目的对象、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 Target			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
1～7			打开最近打开过的项目

4）调试菜单 Debug 和调试命令（如表 1-3 所示）

表 1-3　　调试菜单和调试命令 Debug

菜单	工具栏	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，遇到子程序则进入
Step over		F10	单步执行程序，跳过子程序
Step out of		Ctrl+F11	执行到当前函数的结束
Current function stop Runing		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/禁止程序运行轨迹的标识
View Trace Records			显示程序运行过的指令
Memory Map…			打开存储器空间设置对话框
Performance Analyzer…			打开设置性能分析的窗口
Inline Assembly…			对某一行重新汇编，可以修改汇编代码
Function Editor…			编辑调试函数和调试设置文件

5）外围器件菜单 Peripherals（如表 1-4 所示）

表 1-4　　外围器件菜单 Peripherals

菜单	工具栏	描述
Reset CPU		复位 CPU
以下为单片机外围器件的设置对话框（对话框的种类及内容依赖于你选择的 CPU）		
Interrupt		中断观察
I/O-Ports		I/O 口观察
Serial		串口观察
Timer		定时器观察
A/D Conoverter		A/D 转换器
D/A Conoverter		D/A 转换器
I^2C Conoverter		I^2C 总线控制器
Watchdog		看门狗

6）工具菜单 Tool（如表 1-5 所示）

利用工具菜单，可以设置并运行 Gimpel PC-Lint、Siemens Easy-Case 和用户程序。通过 Customize Tools Menu…菜单，可以添加需要的程序。

表 1-5　　工具菜单 Tool

菜单	描述
Setup PC-Lint…	设置 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 处理项目中所有的 C 源代码文件
Setup Easy-Case…	设置 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

3．创建项目实例

μVision2 包括一个项目管理器，它可以使 8x51 应用系统的设计变得简单。要创建一个应用，需要按下列步骤进行操作：

- 启动 μVision2，新建一个项目文件并从器件库中选择一个器件。
- 新建一个源文件并把它加入到项目中。
- 增加并设置选择的器件的启动代码
- 针对目标硬件设置工具选项。
- 编译项目并生成可编程 PROM 的 HEX 文件。

下面将逐步地进行描述，从而指引读者创建一个简单的 μVision2 项目。

1）选择【Project】/【New Project】选项，如图 1-2 所示。

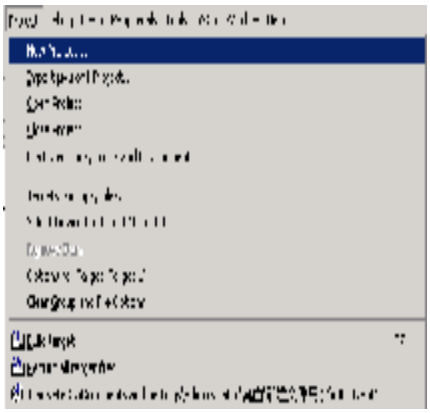


图 1-2　Project 菜单

2）在弹出的“Create New Project”对话框中选择要保存项目文件的路径，比如保存到 Exercise 目录里，在“文件名”文本框中输入项目名为 example，如图 1-3 所示，然后单击“保存”按钮。

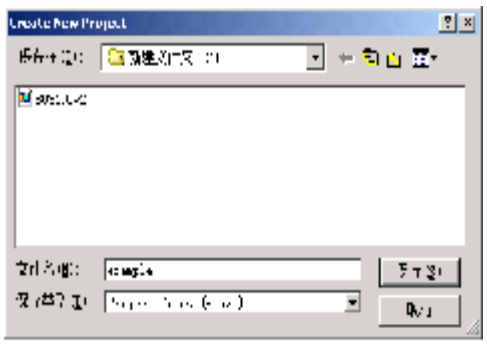


图 1-3　Create New Project 对话框

3）时会弹出一个对话框，要求选择单片机的型号。读者可以根据使用的单片机型号来选择，Keil C51 几乎支持所有的 51 核的单片机，这里只是以常用的 AT89C51 为例来说明，如图 1-4 所示。选择 89C51 之后，右边 Description 栏中即显示单片机的基本说明，然后单击“确定”按钮。

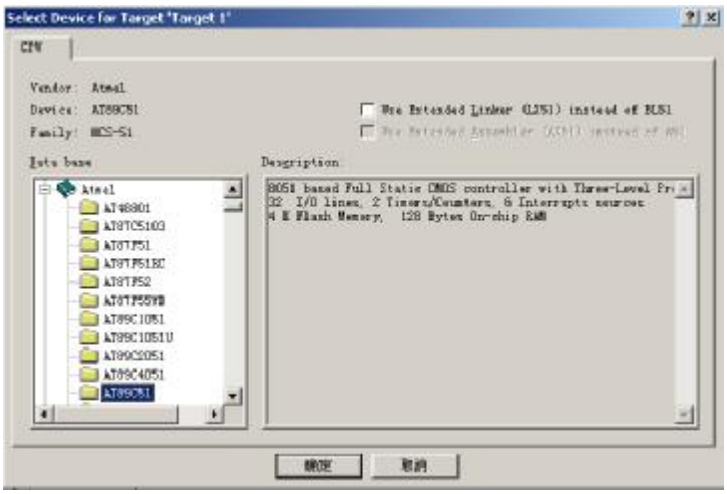


图 1-4 选择单片机的型号对话框

4）这时需要新建一个源程序文件。建立一个汇编或 C 文件，如果已经有源程序文件，可以忽略这一步。选择【File】/【New】选项，如图 1-5 所示。

5）在弹出的程序文本框中输入一个简单的程序，如图 1-6 所示。

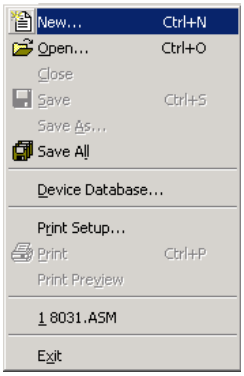


图 1-5　新建源程序文件对话框图

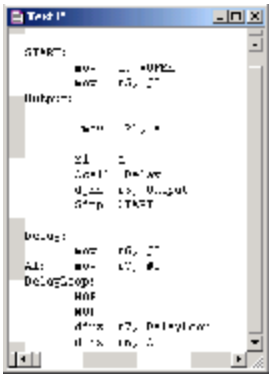


图 1-6　程序文本框

6）选择【File】/【Save】选项，或者单击工具栏  按钮，保存文件。

在弹出的如图 1-7 所示的对话框中选择要保存的路径，在“文件名”文本框中输入文件名。注意一定要输入扩展名，如果是 C 程序文件，扩展名为.c；如果是汇编文件，扩展名为.asm；如果 ini 文件，扩展名为. ini。这里需要存储 ASM 源程序文件，所以输入.asm 扩展名（也可以保存为其他名字，比如 new.asm 等），单击“保存”按钮。

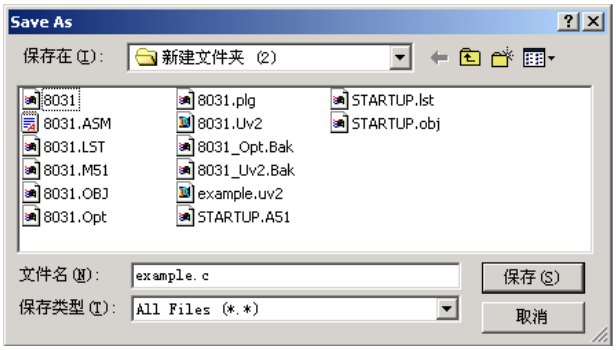


图 1-7　“Save As”对话框图

7）单击 Target1 前面的+号，展开里面的内容 Source Group1，如图 1-8 所示。

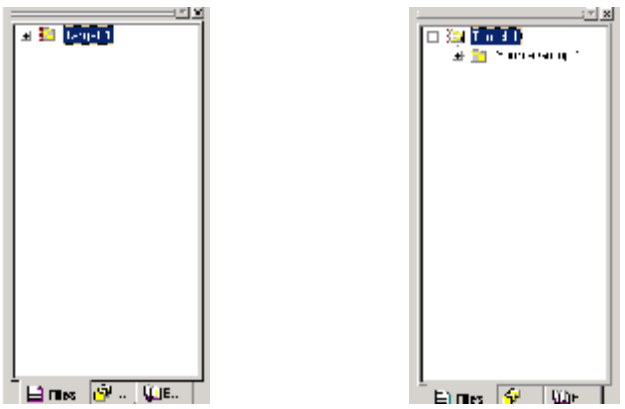


图 1-8　Target 展开图

8）用右键单击 Source Group1，在弹出的快捷菜单中选择 Add File to Group`Source Group1`选项，如图 1-9 所示。

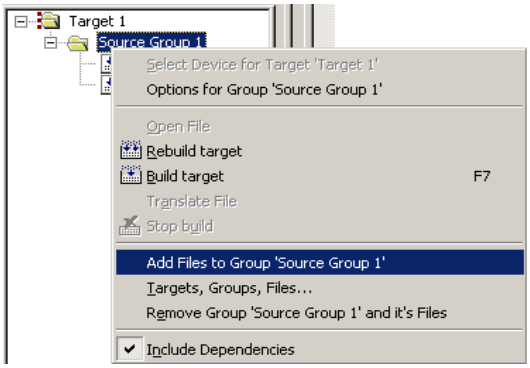


图 1—9 Add Files to Group ‘Source Group1’菜单

9）选择刚才的文件 example.asm，文件类型选择 Asm Source file（*.C）。如果是 C 文件，则选择 C Source file；如果是目标文件，则选择 Object file；如果是库文件，则选择 Library file。最后单击“Add”按钮，如果要添加多个文件，可以不断添加。添加完毕后单击“Close”按钮，关闭该窗口，如图 1-10 所示

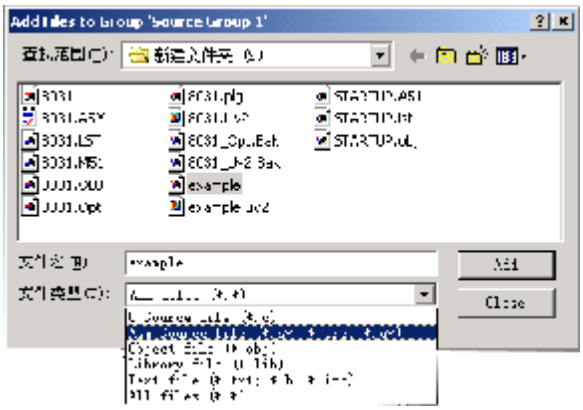


图 1-10 Add Files to Group ‘Source Group1’对话框

10）这时在 Source Group1 目录里就有 example.asm 文件，如图 1-11 所示。

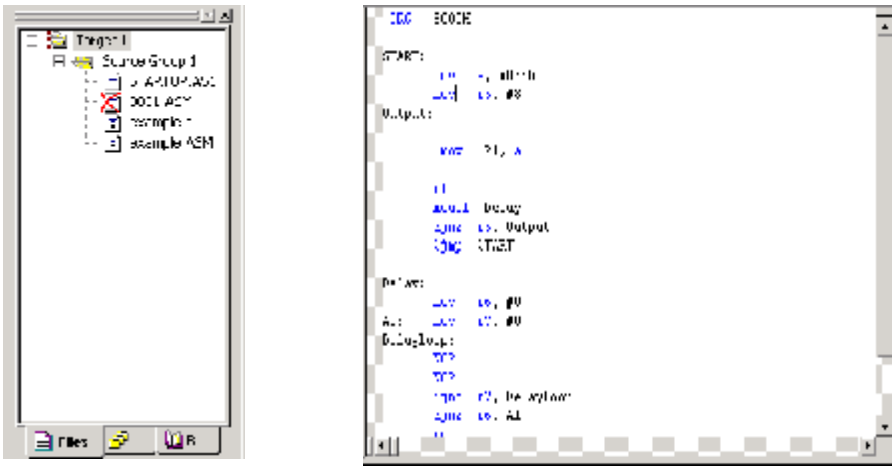


图 1-11 example.asm 文件

11）接下来要对目标进行一些设置。用鼠标右键（注意用右键）单击 Target1，在弹出的会计菜单中选择 Options for Target “Target 1”选项，如图 1-12 所示。

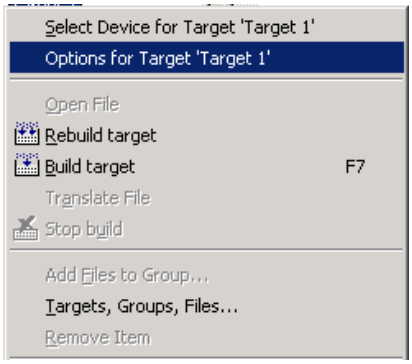
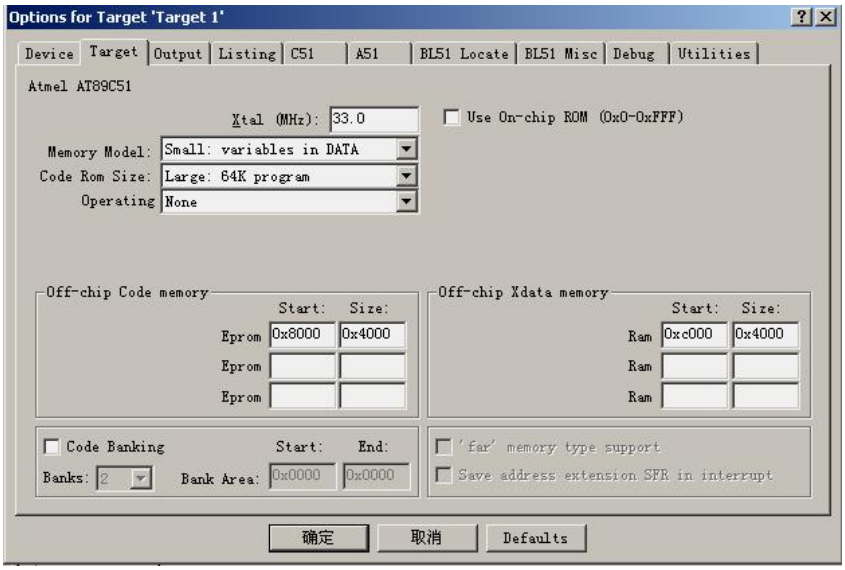


图 1-12 ptions for Target “Target 1”选项

12）弹出 Options for Target “Target 1”对话框，其中有 8 个选项卡。

① 默认为 Target 选项卡（如图 1-13 所示）



1-13 Target 选项卡

- Xtal（MHZ）：设置单片机工作的频率，默认是 24.0MHz。
- Use On-chip ROM(0x0-0XFFF)：表示使用片上的 Flash ROM，At89C51 有 4KB 的可重编程的 Flash ROM，该选项取决于单片机应用系统，如果单片机的 EA 接高电平，则选中这个选项，表示使用内部 ROM，如果单片机的 EA 接低电平，表示使用外部 ROM，则不选中该项。这里选中该选项。
- Off-chip Code memory：表示片外 ROM 的开始地址和大小，如果没有外接程序存储器，那么不需要填任何数据。这里假设使用一个片外 ROM，地址从 0x8000 开始，一般填 16 进制的数，Size 为片外 ROM 的大小。假设外接 ROM 的大小为 0x1000 字节，则最多可以外接 3 块 ROM。
- Off-chip Xdata memory：那么可以填上外接 Xdata 外部数据存储器的起始地址和大小，一般的应用是 62256，这里特殊的指定 Xdata 的起始地址为 0x2000，大小为 0x8000。
- Code Banking：是使用 Code Banking 技术。Keil 可以支持程序代码超过 64KB 的情况，

最大可以有 2MB 的程序代码。如果代码超过 64KB，那么就要使用 Code Banking 技术，以支持更多的程序空间。Code Banking 支持自动的 Bank 的切换，这在建立一个大型系统时是必需的。例如：在单片机里实现汉字字库，实现汉字输入法，都要用到该技术。

- Memory Model：单击 Memory Model 后面的下拉箭头，会有 3 个选项，如图 1-14 所示。

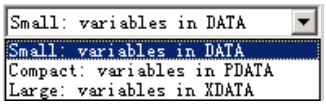


图 1-14　Memory Model 选项

- Small：变量存储在内部 RAM 里。
- Compact：变量存储在外部 RAM 里，使用 8 位间接寻址。
- Large：变量存储在外部 RAM 里，使用 16 位间接寻址。

一般使用 Small 来存储变量，此时单片机优先将变量存储在内部 RAM 里，如果内部 RAM 空间不够，才会存在外部 RAM 中。Compact 的方式要通过程序来指定页的高位地址，编程比较复杂，如果外部 RAM 很少，只有 256 字节，那么对该 256 字节的读取就比较快。

如果超过 256 字节，而且需要不断地进行切换，就比较麻烦，Compact 模式适用于比较少的外部 RAM 的情况。Large 模式是指变量会优先分配到外部 RAM 里。需要注意的是，3 种存储方式都支持内部 256 字节和外部 64KB 的 RAM。因为变量存储在内部 RAM 里运算速度比存储在外部 RAM 要快得多，大部分的应用都是选择 Small 模式。

使用 Small 模式时，并不说明变量就不可以存储在外部，只是需要特别指定，比如：

unsigned char xdata a：变量 a 存储在内部 RAM。

unsigned char a：变量存储在内部 RAM。

但是使用 Large 的模式时：

unsigned char xdata a：变量 a 存储在外部 RAM。

unsigned char a：变量 a 同样存储在外部 RAM。

这就是它们之间的区别，可以看出这几个选项只影响没有特别指定变量的存储空间的情况，默认存储在所选模式的存储空间，比如上面的变量定义 unsigned char a。

- Code Rom Size：单击 Code Rom Size 后面的下拉箭头，将有 3 个选项，如图 1-15 所示。

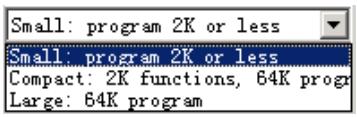


图 1-15　Code Rom Size 选项

- Small：program2K or less，适用于 AT89C2051 这些芯片，2051 只有 2KB 的代码空间，所以跳转地址只有 2KB，编译的时候会使用 ACALL AJMP 这些短跳指令，而不会使用 LCALL，LJMP 指令。如果代码地址跳转超过 2KB，那么会出错。
- Compact：2K functiongs，64K program，表示每个子函数的代码大小不超过 2K，整个项目可以有 64K 的代码。就是说在 main()里可以使用 LCALL，LJMP 指令，但在子程序里只会使用 ACALL，AJMP 指令。只有确定每个子程序不会超过 2KB，才可以使用 Compact 方式。
- Large：64KB program，表示程序或子函数代码都可以大到 64KB，使用 code bank 还可以更大。通常都选用该方式。选择 Large 方式速度不会比 Small 慢很多，所以一般没有必要

选择 Compact 和 Small 方式。这里选择 Large 方式。

l Operating：单击 Operating 后面的下拉箭头，会有 3 个选项，如图 1-16 所示。

图 1-16 Operating 选项

Ÿ None：表示不使用操作系统。

Ÿ RTX-51 Tiny Real-Time OS：表示使用 Tiny 操作系统。

Ÿ RTX-51 Full Real -Time OS：表示使用 Full 操作系统。

Tiny 是一个多任务操作系统，使用定时器 0 做任务切换。在 11.0592MHz 时，切换任务的速度为 30ms。如果有 10 个任务同时运行，那么切换时间为 300ms。不支持中断系统的任务切换，也没有优行级，因为切换的时间太长，实时性大打折扣。多任务情况下（比如 5 个），轮循一次需要 150ms，即 150ms 才处理一个任务，这连键盘扫描这些事情都实现不了，更不要说串口接收、外部中断了。同时切换需要大概 1000 个机器周期，对 CPU 的浪费很大，对内部 RAM 的占用也很严重。实际上用到多任务操作系统的情况很少。

Keil C51 Full Real -Time OS 是比 Tiny 要好一些的系统（但需要用户使用外部 RAM），支持中断方式的多任务和任务优先级，但是 Keil C51 里不提供该运行库，要另外购买。

这里选择 None。

②设置 Output 选项卡（如图 1-17 所示）

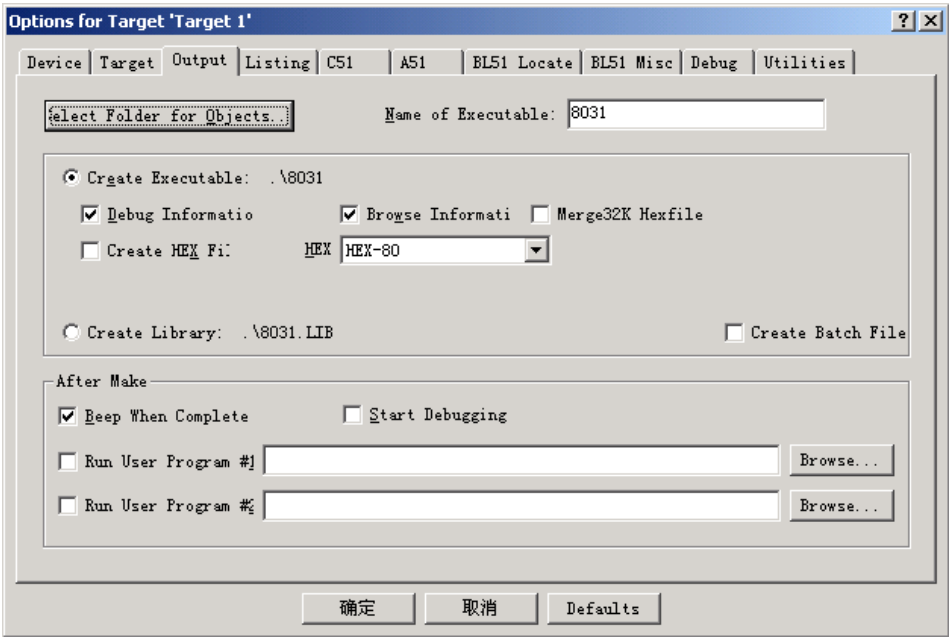


图 1-17 设置 Output 卡

l Select Folder for Objects：单击该按钮可以选择编译后目标文件的存储目录，如果不设置，就存储在项目文件的目录里。

l Name of Executable：设置生成的目标文件的名字，缺省情况下和项目的名字一样。目标文件可以生成库或者 obj、HEX 的格式。

■ Create Executable：如果要生成 OMF 以及 HEX 文件，一般选中 Debug Information 和 Browse Information。选中这两项，才有调试所需的详细信息，比如要调试 C 语言程序，如果不选中，调试时将无法看到高级语言写的程序。

■ Create HEX File：要生成 HEX 文件，一定要选中该选项，如果编译之后没有生成 HEX 文件，就是因为这个选项没有被选中。默认是不选中的。

■ Create Library：选中该项时将生成 lib 库文件。根据需要决定是否要生成库文件，一般应用是不生成库文件的。

■ After Make：栏中有以下几个设置。

■ Beep when complete：编译完成之后发出咚的声音。

■ Start Debugging：马上启动调试（软件仿真或硬件仿真），根据需要来设置，一般是不选中。

■ Run User Program #1，Run User Program #2：这个选项可以设置编译完之后所要运行的其他应用程序（比如有些用户自己编写了烧写芯片的程序，编译完便执行该程序，将 HEX 文件写入芯片），或者调用外部的仿真器程序。根据自己的需要设置。

③设置 Listing 选项卡（如图 1-18 所示）

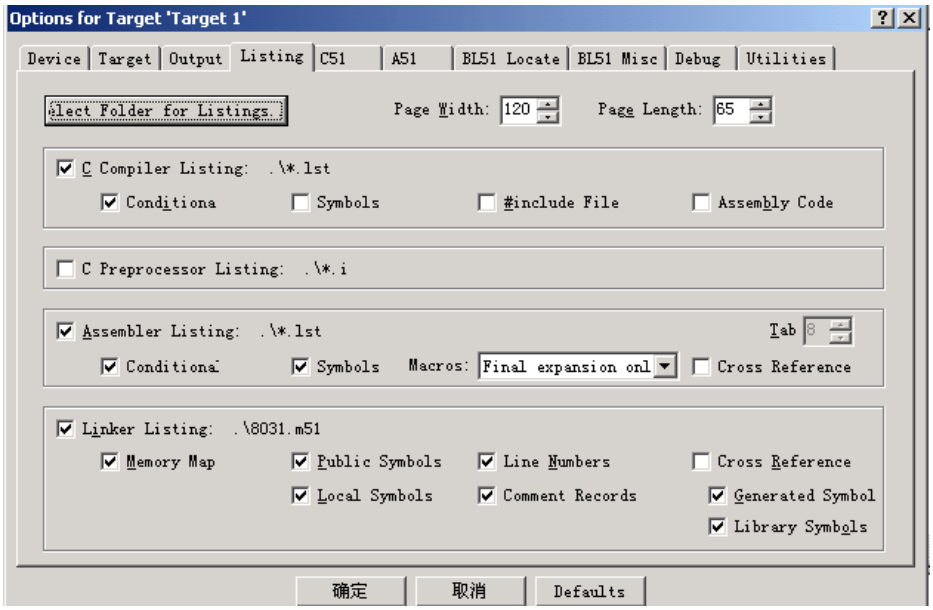


图 1-18　设置 Listing 选项卡

Keil C51 在编译之后除了生成目标文件之外，还生*.lst、*m51 的文件。这两个文件可以告诉程序员程序中所用的 idata、data、bit、xdata、code、RAM、ROM、stack 等的相关信息，以及程序所需的代码空间。

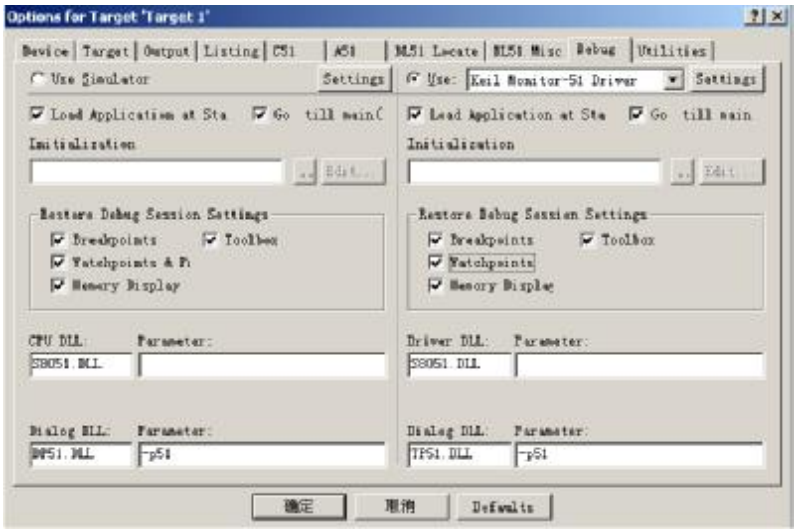
选中 Assembly Code 会生成汇编的代码。这是很有好处的，如果不知道如何用汇编来写一个 long 型数的乘法，那么可以先用 C 语言来写，写完之后编译，就可以得到用汇编实现的代码。对于一个高级的单片机程序员来说，往往既要熟悉汇编，同时也要熟悉 C 语言，才能更好地编写程序。某些地方用 C 语言无法实现，便用汇编语言却很容易。有些地方用汇编语言，很繁琐，用 C 语言就很方便。

单击 Select Folder for Listings 按钮后，在出现的对话框中可以选择生成的列表文件的存放目录。不做选择时，使用项目文件所在的目录。

④**设置 Debug 选项卡**（如图 1-19 所示）

这里有两类仿真形式可选：Use Simulator 和 Use：Keil Monitor-51 Driver，前一种是纯软件仿真，后一种是带有 Monitor-51 目标仿真器的仿真。

- Load Application at Start：选择这项之后，Keil 才会自动装载程序代码。



1-19 设置 Debug 选项卡

- Go till main：调试 C 语言程序时可以选择这一项，PC 会自动运行到 main 程序处。

这里选择 Use Simulator。

如果选择 Use：Keil Monitor-51 Driver，还可以单击图 1-19 中的 Settings 按钮，打开新的窗口如图 1-20，其中的设置如下。

- Port：设置串口号，为仿真机的串口连接线 COM_A 所连接的串口。
- Baudrate：设置为 9600，仿真机固定使用 9600bit/s 跟 Keil 通信。
- Serial Inerrupt：允许串行中断，选中它。
- Cache Options：可以选也可以不选，推荐选它，这样仿真机会运行得快一点。

最后单击 OK 按钮关闭窗口。

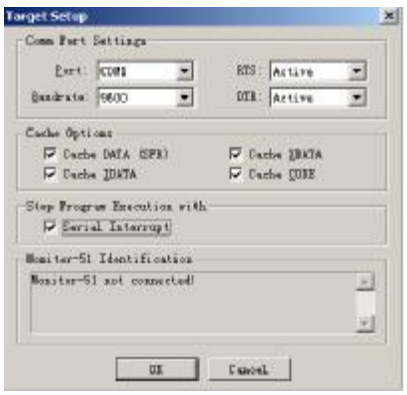


图 1-20 Target 设置

13）编译程序，选择【Project】/【Rebuild all target files】选项，如图 1-21 所示。

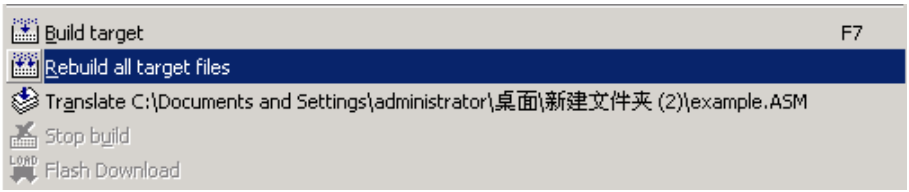


图 1-21　Rebuild all target files

或者单击工具栏中的按钮，如图 1-22 所示，开始编译程序。

图 1-22　工具栏中的按钮

如果编译成功，开发环境下面会显示编译成功的信息，如图 1-23 所示。

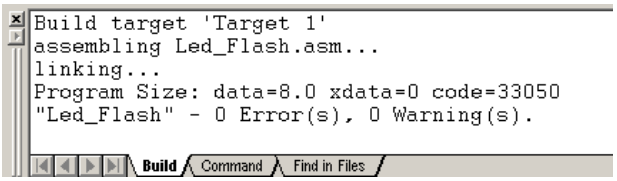


图 1-23　编译成功信息

14）编译完毕之后，选择【Debug】/【Start/Stop Debug Session】选项，即就进入仿真环境，如图 1-24 所示。

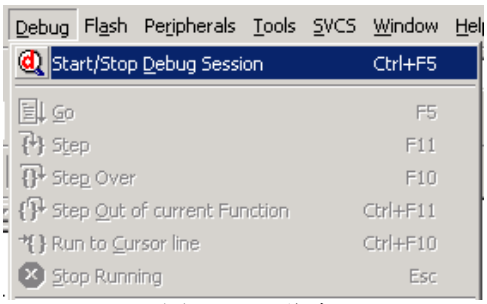


图 1-24　仿真

或者单击工具栏中的铵钮，如图 1-25 所示。

图 1-25　工具栏仿真按钮

15）装载代码之后，开发环境下面显示如图 1-26 所示的信息。

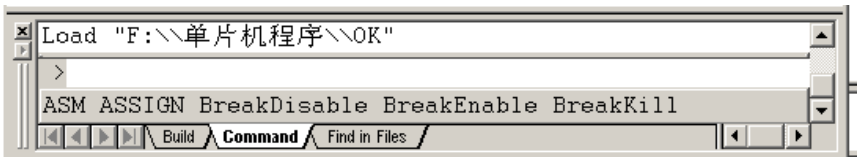


图 1-26　装载代码

目录