第 1 章　单片机硬件基础

单片微型计算机简称为单片机。单片机在一块芯片上集成了中央处理器(CPU)、存储器(数据存储器 RAM、程序存储器 ROM)、定时/计数器和 I/O 端口等主要部件。常见的 51 系列单片机有 4 个 8 位的双向并行输入/输出(I/O)端口(P0 口、P1 口、P2 口、P3 口)，每个端口既可以按字节进行输入、输出，也可以按位输入、输出高/低电平。利用单片机的 I/O 端口可以方便地实现单片机与外围数字设备或芯片之间的信息传递。

1.1　51 单片机芯片引脚

单片机芯片的封装有直插式封装(DIP)与表面贴片式封装(SMD)两种，本书以直插式封装的单片机为例进行介绍。图 1.1 所示为 DIP 封装的 STC89C52 单片机引脚图。

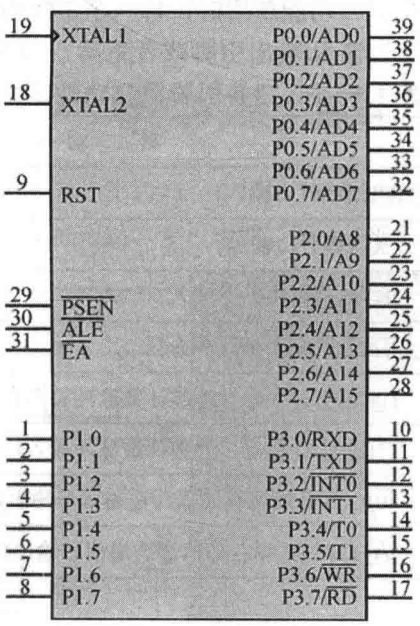

图 1.1　DIP 封装的 STC89C52 单片机引脚图

单片机有 40 个引脚，分为电源线、端口线和控制线三类。

1. 电源线

(1) Vss (20 脚)：接地引脚。

(2) Vcc (40 脚)：正电源引脚。正常工作时，接 +5 V 电源。

2. 端口线

51系列单片机片内有4个8位并行I/O端口，即P0、P1、P2、P3口。它们可以双向使用。

(1) P0口。如图1.1所示，32～39脚为P0.0～P0.7输入/输出引脚。P0口是一个双向的8位并行I/O口，每个I/O口可独立控制，片内没有上拉电阻，输入为高阻态，所以不能正常输出高/低电平，因此，P0端口在使用中需要外接上拉电阻，方可输出高/低电平。如图1.2所示，一般上拉电阻选择10kΩ电阻。P0端口的驱动能力为其他端口(P1、P2、P3)的2倍。

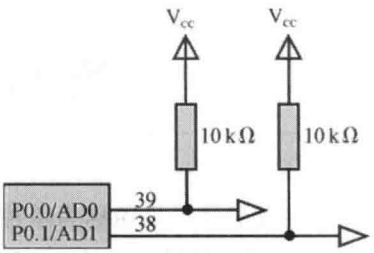

图1.2　上拉电阻

(2) P1口。1～8脚为P1.0～P1.7输入/输出引脚。P1口是一个准双向的8位并行I/O口，每个I/O口可独立控制，内部具有上拉电阻，故能正常输出高/低电平。I/O口在作为输入口时，须先输出高电平作为准备，所以称为准双向口。

(3) P2口。21～28引脚为P2.0～P2.7输入/输出引脚。P2口是一个准双向的8位并行I/O口，每个I/O口可独立控制，内部具有上拉电阻，与P1口相似。

(4) P3口。10～17脚为P3.0～P3.7输入/输出引脚。P3口是一个准双向的8位并行I/O口，每个I/O口可独立控制，内部具有上拉电阻。P3口作为第一功能使用时就是普通的I/O口，与P1口相同。P3口作为第二功能使用时，每一个I/O引脚的定义如表1.1所示。P3口的每一个引脚可以单独定义为输入/输出引脚或者是第二功能引脚。

表1.1　P3口各引脚第二功能定义

P3口引脚	第二功能
P3.0	RXD(串行口输入)
P3.1	TXD(串行口输出)
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/计数器0外部计数脉冲输入)
P3.5	T1(定时/计数器1外部计数脉冲输入)
P3.6	$\overline{\text{WR}}$(片外数据存储器写选通信号输出)
P3.7	$\overline{\text{RD}}$(片外数据存储器读选通信号输出)

3. 控制线

(1) RST(9引脚)。RST为单片机的复位引脚。当引脚上出现24个时钟周期以上的高电平时有效。复位后，单片机程序重新开始执行，单片机正常工作时，该引脚应保持低电平。

(2) XTAL1和XTAL2(19，18引脚)。XTAL1引脚为片内振荡电路的输入端，XTAL2引脚为片内振荡电路的输出端。51系列单片机的时钟产生有两种方式，一种是内时钟振荡方式(如图1.3左图)，需要在18和19引脚上外接石英晶体和振荡电容，一种是外部时钟振荡方式，即将XTAL1接地，外部时钟信号从XTAL2脚输入(如图1.3右图)。

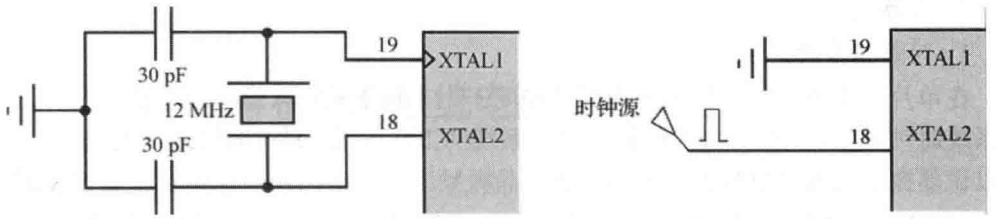

图 1.3　内、外时钟连接方式

(3) $\overline{\text{ALE}}$/PROG (30 引脚)。

$\overline{\text{ALE}}$/PROG 引脚为地址锁存允许/编程引脚。当访问外部程序存储器时，$\overline{\text{ALE}}$/PROG 的输出用于锁存地址的低位字节。当不访问外部程序存储器时，$\overline{\text{ALE}}$/PROG 端将输出一个 1/6 时钟频率的正脉冲信号，这个信号可以用于识别单片机是否工作，也可以当做一个时钟向外输出。

(4) $\overline{\text{EA}}$/Vpp(31 引脚)。$\overline{\text{EA}}$/Vpp 引脚允许访问片外程序存储器/编程电源线。该引脚接高电平时，访问片内程序存储器；该引脚接低电平时，则访问片外程序存储器，即

$\overline{\text{EA}}$ = 1，片内程序存储器有效；

$\overline{\text{EA}}$ = 0，片外程序存储器有效，此时必须有外部扩展存储器。

通常在使用中，该脚接高电平。

(5) $\overline{\text{PSEN}}$ (29 引脚)。$\overline{\text{PSEN}}$ 为片外 ROM 选通线。

1.2　单片机最小系统

单片机最小系统是指用最少的元件组成的一个可以工作的应用系统，对于 51 系列单片机来讲，最小系统主要包括单片机、晶振电路、复位电路。图 1.4 所示为单片机的最小系统原理图。

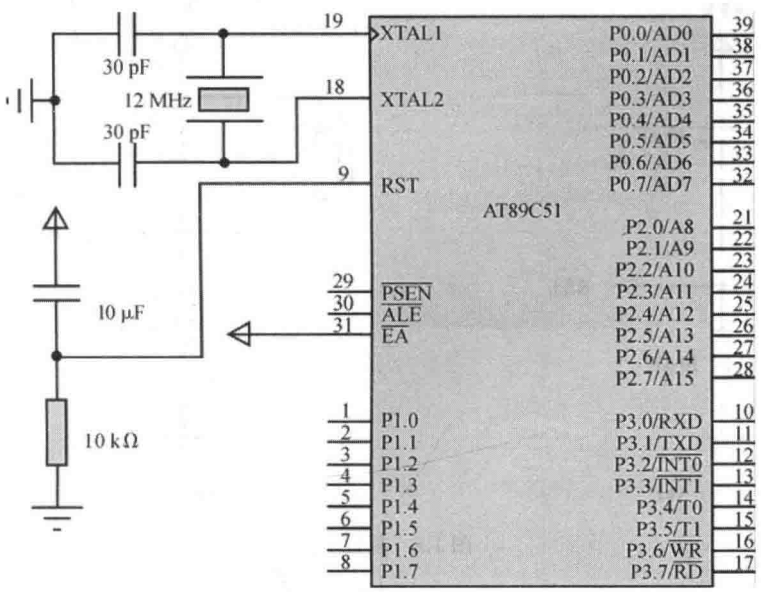

图 1.4　单片机最小系统原理图

1. 晶振电路

1) 时钟信号的产生

在单片机系统中，所有的工作都是在同一个节拍下进行，这样才不会冲突，这个节拍由系统时钟产生。系统时钟的快慢决定了系统的工作效率，系统时钟通常由晶振电路提供，可以说晶振电路就是单片机系统的心脏。晶振频率大小由用户自己确定，以 STC89C52RC 增强型 8051 单片机为例，可接晶振频率为 0～40 MHz，推荐值为 11.0592 MHz、12 MHz，振荡电容的大小一般取 10～30 pF，推荐值为 30 pF。

2) 时序

(1) 时钟周期。时钟周期又称为振荡周期，由单片机的内部振荡电路 OSC 产生，定义为 OSC 时钟频率的倒数，即 $T_{时}=1/f_{osc}$。时钟频率的大小由晶振的大小决定。

(2) 机器周期。机器周期为单片机的基本操作周期，在一个机器周期内，CUP 可以完成一个最简单的独立操作。MQ51 单片机的一个机器周期由 12 个时钟周期组成，即机器周期 = 12*时钟周期。例如，若单片机系统的振荡器频率为 12 MHz，则可以计算出 1 个机器周期的时间为 1 μs。

3) 电磁兼容性的考虑

时钟源通常是系统中最严重的电磁辐射源，如果系统中接线过长，该长线就变成了天线，因此，在布线的时候要求时钟源尽量靠近单片机，布线尽量短。

2. 复位电路

MCS-51 单片机有一个复位引脚 RST(9 脚)，高电平有效。在时钟电路工作以后，当外部电路使得该引脚上出现两个机器周期(24 个时钟周期)以上的高电平时，单片机复位。常用的复位有两种方式：上电复位(对应电路如图 1.5(a)所示)和手动复位(对应电路如图 1.5(b)所示)。

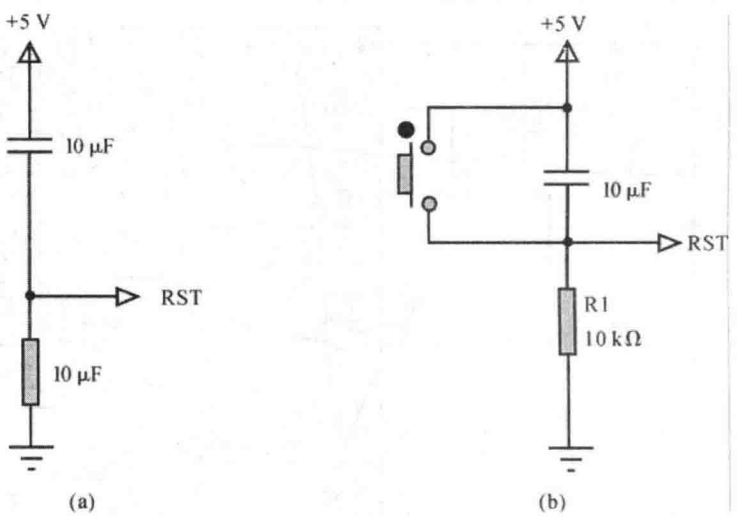

图 1.5　复位电路

注意：单片机复位后，P0～P3 输出都为高电平。

1.3　单片机最小系统电路设计应注意的问题

设计单片机最小系统电路时需要注意以下问题：

(1) P0 口需要加上上拉电阻，推荐用 10 kΩ的排阻。

(2) 在 P0、P1、P2、P3 各个端口外面加上排针，方便接线用。

(3) 在电源输入端加入 104 滤波电容。

(4) $\overline{EA}$引脚直接接到 V_{cc}。

(5) 要多加电源接线针。

(6) 设计四个下载程序用的接口针，接口针分别连接到 V_{cc}、GND、P3.0、P3.1。

(7) 设计 PCB 时，晶振需要靠近单片机的 18、19 引脚，晶振的起振电容不能离晶振过远。

(8) 设计 PCB 时，单片机的四个下载接口应在电路板的边沿，方便接线。

1.4　习　　题

(1) 设 51 单片机的晶振是 12 MHz，则单片机的时钟周期和机器周期是多大?

(2) 51 单片机的起振电容一般是多大?

(3) 51 单片机的引脚有多少个?

(4) 如果 51 单片机要使用片内的程序存储器，则 $\overline{EA}$ 引脚需要接什么电平?

(5) 51 单片机的哪一个端口内部没有上拉电阻?

(6) 51 单片机的哪一个端口有第二功能?

(7) 51 单片机总共有多少个 I/O 口?

(8) 51 单片机的第几引脚是复位引脚?

第2章　单片机开发环境

单片机应用系统的仿真开发平台有两个常用的工具软件：Keil C51 和 Proteus ISIS。Keil C51 是美国 Keil Software 公司出品的 51 系列兼容单片机 C 语言软件开发系统，Keil 用于 C 语言源程序的编辑、编译、链接调试仿真。Proteus 是英国 Lab Center Electronics 公司开发的电路分析与实物仿真软件，Proteus 软件由 ISIS 和 ARES 两个软件构成，其中 ISIS 是原理图编辑与仿真软件，ARES 是布线编辑软件，本文只介绍 Proteus ISIS 软件。

2.1　Keil C 的使用

Keil C51 到目前经历了多个版本，下面通过 Keil μVision 4 版介绍系统的功能和使用。

1. Keil C 的安装

Keil μVision 4 的安装与其他软件安装的方法相同，安装过程比较简单，安装目录按照默认目录即可。

2. Keil μVision 4 界面介绍

单击“Keil μVision4”图标，启动 Keil μVision4 程序，可以看到如图 2.1 所示的 Keil μVision4 主界面。

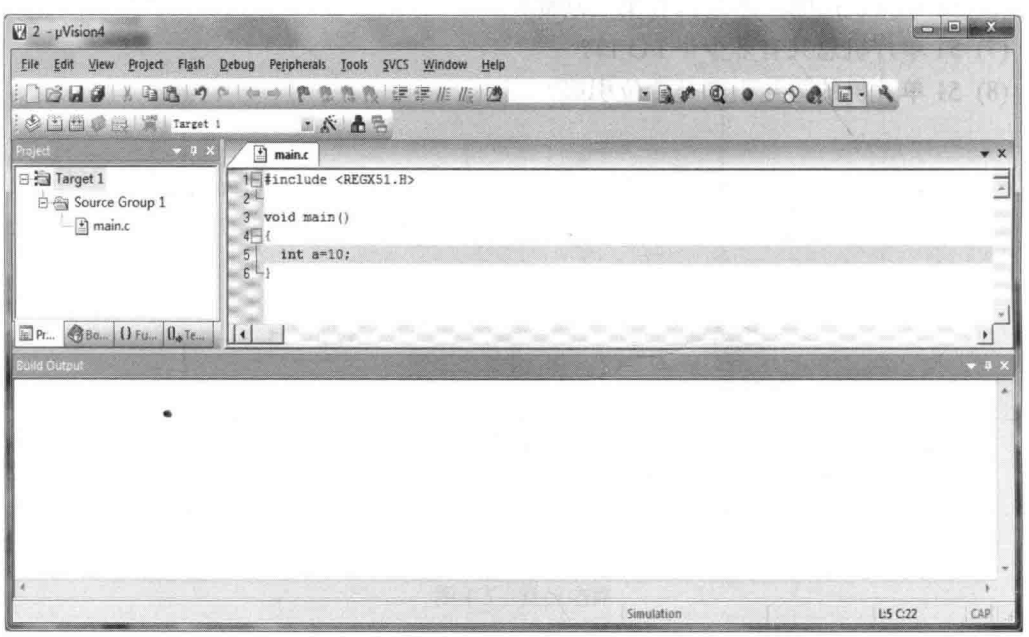

图 2.1　Keil μVision4 的主界面

Keil μVision4 的主界面提供各种操作菜单，如文件操作、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，工具条按钮提供键盘快捷键(用户可自行设置)。下面以表格的形式简要介绍 Keil μVision4 中常用的菜单栏、工具按钮和快捷方式。

Keil μVision4 有两种操作模式：编辑模式和调试模式，通过用“Debug”菜单下的“Start/Stop Debugging”(开始/停止调试模式)命令可以进行切换。编辑模式可以建立项目、文件，编译项目、文件产生可执行的程序；调试模式提供的调试器可以用来调试项目。

(1) 文件菜单(File)：文件菜单说明如表 2.1 所示。

表 2.1　文件菜单说明

File 菜单	工具按钮	快捷键	说　明
New		Ctrl + N	创建一个新的文本文件(源程序文件)
Open		Ctrl + O	打开一个已有的文件
Close			关闭当前文件
Save		Ctrl + S	保存当前文件
Save as…			保存并重新命名当前文件
Save All			保存所有打开的文本文件(源程序文件)
Device Database			维护 μ Vision4 设备数据库
Print Setup			打印机安装
Print		Ctrl + P	打印当前文件
Print Preview			打印预览
Exit			退出 μ Vision4

(2) 编辑菜单(Edit)：编辑菜单的说明如表 2.2 所示。

表 2.2　编 辑 菜 单

Edit 菜单	工具按钮	快捷键	说　明
Undo		Ctrl + Z	撤销上次操作
Redo		Ctrl + Shift + Z	重复上次撤销的操作
Cut		Ctrl + X	将所选文本剪切到剪贴板
Copy		Ctrl + C	将所选文本复制到剪贴板

续表

Edit 菜单	工具按钮	快捷键	说　明
Paste		Ctrl + V	粘贴剪贴板上的文本
Toggle Bookmark		Ctrl + F2	设置/取消当前行的书签
Goto Next bookmark		F2	移动光标到下一个书签
Goto Previous bookmark		Shift + F2	移动光标到上一个书签
Clear All Bookmark			清除当前文件的所有书签
Find		Ctrl + F	在当前文件中查找文本
Replace		Ctrl + H	替换特定的文本
Find in Files			在几个文件中查找文本

(3) 视图菜单(View)。视图菜单的说明如表 2.3 所示。

表 2.3　视 图 菜 单

View 菜单	工具按钮	说　明
Status Bar		显示/隐藏状态栏
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/隐藏工程窗口
Output Window		显示/隐藏输出窗口
Source Brower		显示/隐藏资源浏览器窗口
Disassembly Window		显示/隐藏反汇编窗口
Watch&Call stack indow		显示/隐藏观察和访问堆栈窗口
Memory Window		显示/隐藏存储器窗口

续表

View 菜单	工具按钮	说　明
Code Coverage Window		显示/隐藏代码覆盖窗口
PreformanceAnalyzerWindow		显示/隐藏性能分析窗口
Serial Window #1		显示/隐藏串行窗口 1
Toolbox		显示/隐藏工具箱
Periodic Window Update		运行程序时，周期刷新调试窗口
Workbook Mode		显示/隐藏工作簿窗口的标签
Include Dependencies		显示/隐藏头文件
Options		设置颜色、字体、快捷键选项

(4) 工程菜单(Project)。常用的工程操作工具如表 2.4 所示。

表 2.4　工程操作工具

Project 菜单	工具按钮	快捷键	说　明
New Project			创建一个新工程
Open Project			打开一个已有的工程
Close Project			关闭当前工程
Components Environment, Books…			定义工具系列、包含文件和库文件的路径
Select Device for Target			从设备数据库中选择一个 CPU
Remove Item			从工程中删除一个组或文件
Options for Target/group/file		Alt + F7	设置对象、组或文件的工具选项
Build target		F7	编译链接当前文件并生成应用
Rebuild all target files			重新编译链接所有文件并生成应用
Translate		Ctrl + F7	编译当前文件
Stop build			停止当前的编译链接进程

(5) 调试操作(Debug)：常用的调试工具菜单如表 2.5 所示。

表 2.5　调 试 菜 单

Debug 菜单	工具按钮	快捷键	说　明
Start/Stop Debug Session			启动/停止调试模式
Go			执行程序，直到下一个有效的断点
Step			跟踪执行程序
Step Over			单步执行程序，跳过子程序
Step Out of current Function			执行到当前函数的结束
Run to Cursor line			执行到光标所在行
Stop Running			停止程序运行
Breakpoints			打开断点对话框
Insert/Remove Breakpoint			在当前行插入/清除断点
Enable/Disable Breakpoint			使能/禁止当前行的断点
Disable All Breakpoint			禁止程序中的所有断点
Kill All Breakpoint			清除程序中的所有断点
Show Next Statement			显示下一条执行的语句/指令
View Trace Records			显示以前执行的指令
Enable/Disable Trace…			使能/禁止程序运行跟踪记录
Memory Map			打开存储器空间配置对话框
Performance Analyzer			打开性能分析器的设置对话框
Inline Assembly			对某一行汇编，可以修改汇编
Function Editor			编辑调试函数和调试配置文件

3. Keil μVision4 工程创建方法

Keil μVision4 是一个集工程管理、源代码编辑、程序调试仿真于一体的集成开发环境。可以用来编写及编译 C 源码、汇编代码，连接和生成目标文件，即 HEX 文件，并且可以调试程序。

一般操作步骤如下：

(1) 创建工程文件。

(2) 给工程添加程序文件(.C 文件或者.ASM 文件)。

(3) 编译程序文件，连接项目，生成 HEX 文件。

(4) 仿真运行、调试，观察结果。

① 启动“Keil μVision4 IDE”后，Keil μVision4 总是打开用户上一次处理的工程，重新建立一个新的工程。建立新工程可以通过执行菜单命令“Project→New μ Vision Project”来实现。如图 2.2 所示。

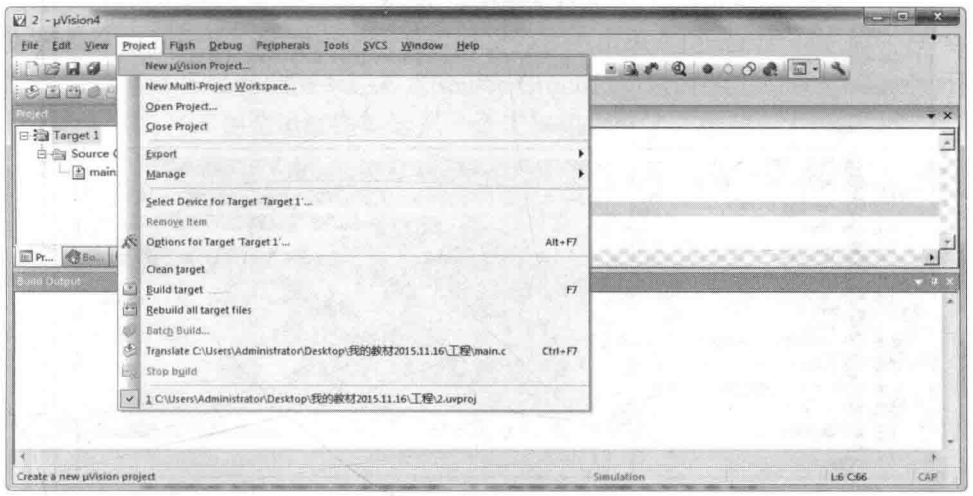

图 2.2　建立新工程界面

② 为工程选择一个存放的目录并取一个名字，建议每个工程单独建立一个目录存放，并将工程中所需要的文件都放在这个目录下。名字可以用中文，建议文件名为“MyProject”，保存类型为默认，最后点击保存。如图 2.3 所示。

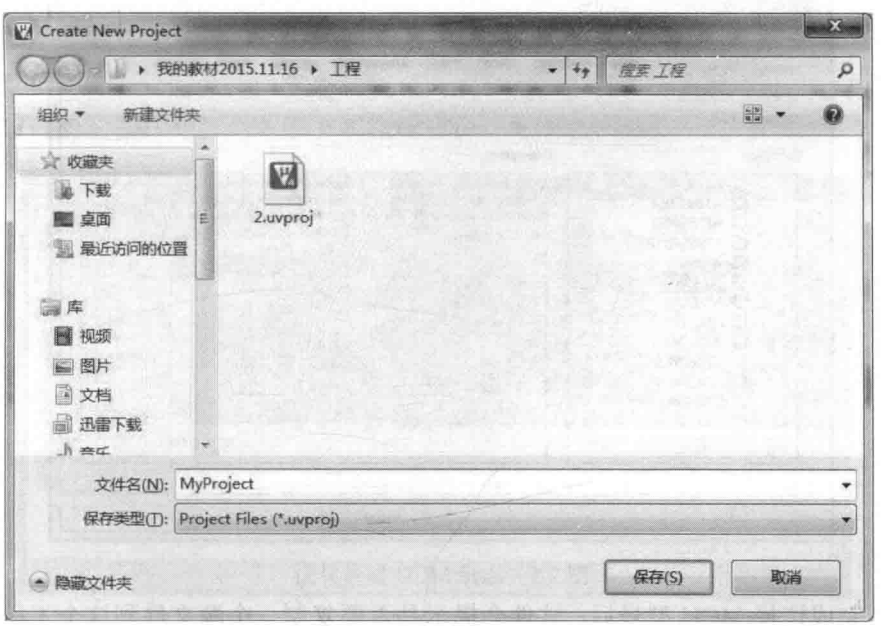

图 2.3　建立存放目录界面

③ 为工程选择目标设备。如图 2.4 所示，这个对话框要求选择目标 CPU (即所用芯片的型号)。Keil 支持的 CPU 很多，以选择 Atmel 公司的 AT89S52 芯片为例。点击“ATMEL”前面的“+”号，展开该层，点击其中的“AT89S52”，如图 2.5 所示，然后再点击“OK”按钮，完成选择 MCU 型号的操作。

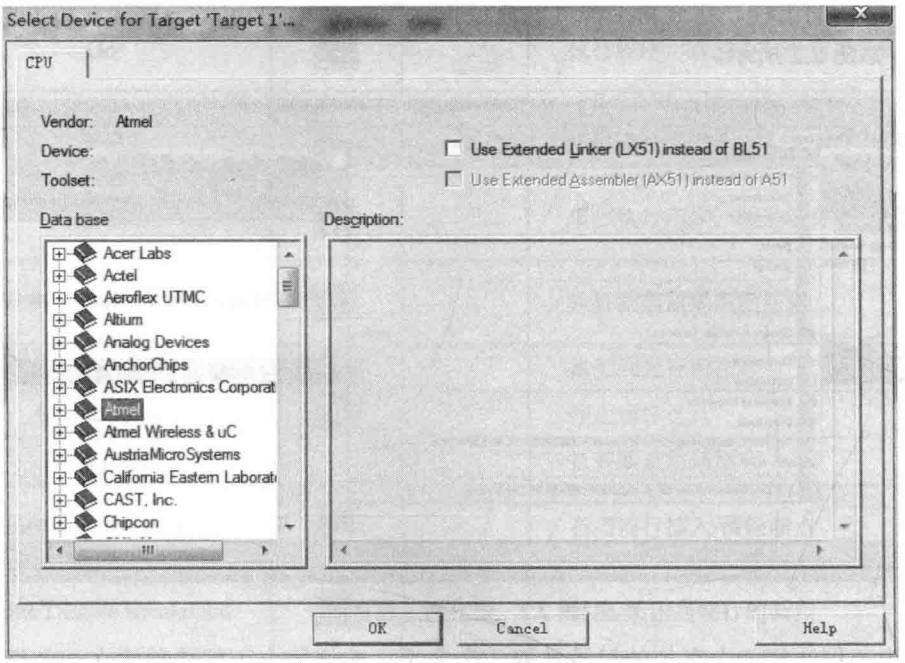

图 2.4　选择目标 CPU 界面

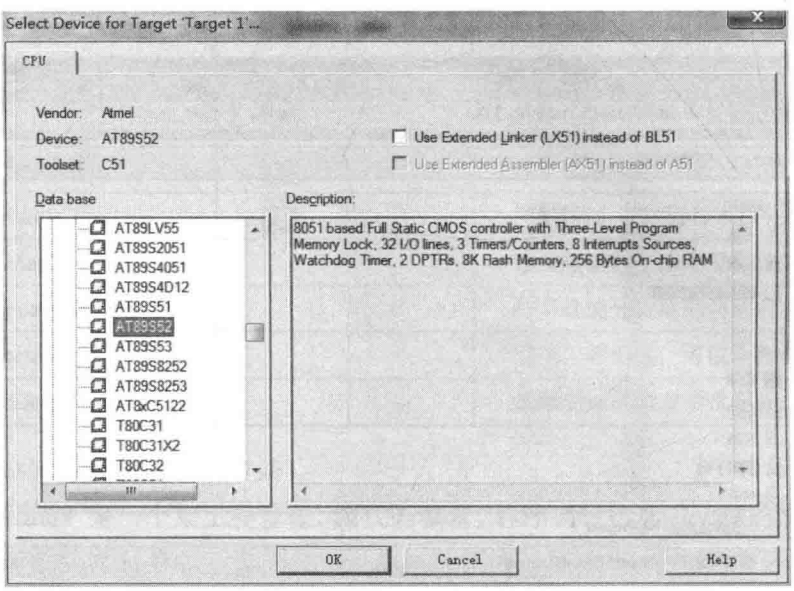

图 2.5　选择 MCU 型号界面

④ 在完成选择 MCU 型号后，软件会提示是否要复制一个源文件到这个工程中，这里我们选择“否”，因为我们要自己添加一个 C 语言或者汇编语言源文件，如图 2.6 所示。

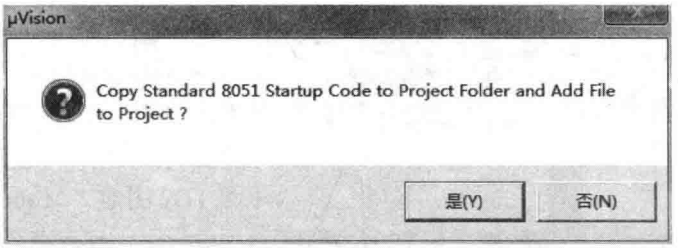

图 2.6　软件提示界面

⑤ 在执行上一步后，就能在工程窗口的文件页中出现“Target 1”了，前面有“+”号，点击“+”号展开，可以看到下一层的“Source Group1”，这时的工程还是一个空的工程，里面什么文件也没有。到这里就完整地把一个工程建立好了，如图 2.7 所示。

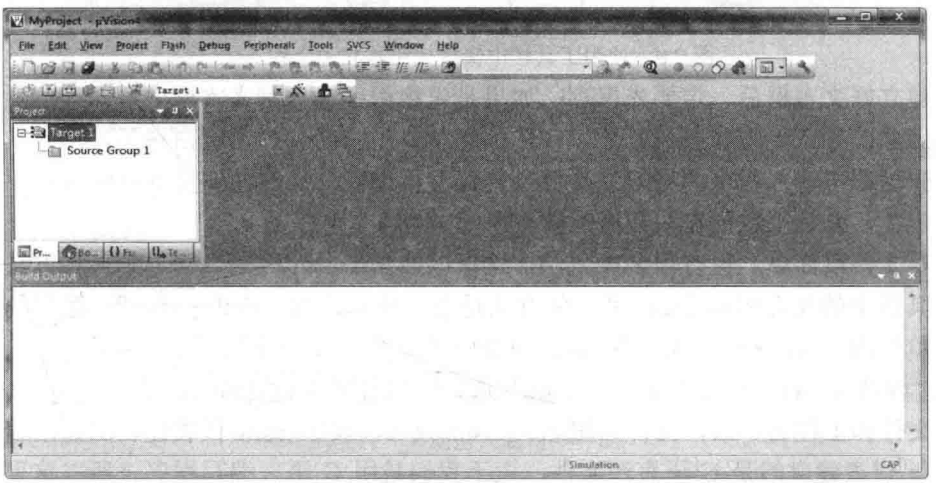

图 2.7　建立好完整工程的界面

4. 源文件的建立

使用菜单“File→New”，如图 2.8 所示，或者点击工具栏的新建文件快捷按钮，就可以在项目窗口的右侧打开一个新的文本编辑窗口，如图 2.9 所示。

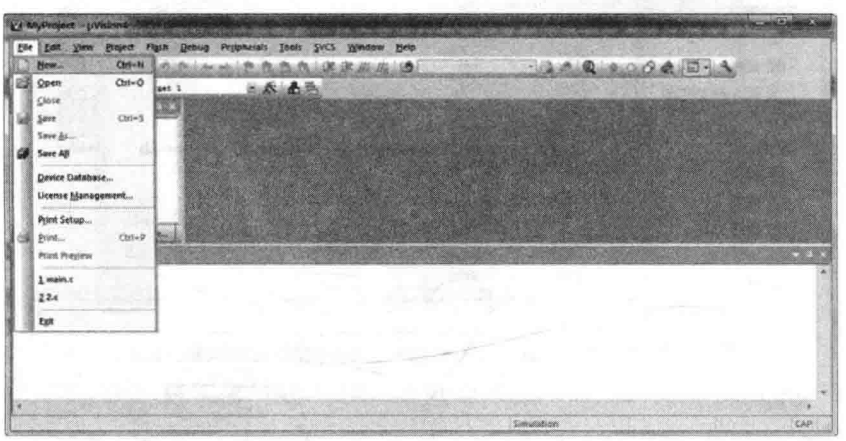

图 2.8　使用菜单建立源文件界面

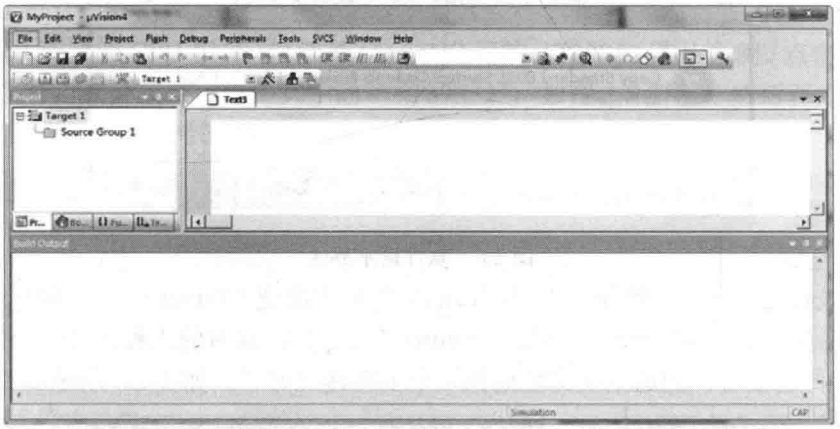

图 2.9　使用工具栏中快捷按钮建立源文件界面

在建立好文本框后一定要先保存，如果是先将程序输入到文本框中再保存的话，有时由于特殊原因导致电脑断电或者死机，那么所花费的时间和精力就白费了，因此我们一定要养成先保存再输入程序的好习惯。而且先保存再输入程序时，在文本框中关键字就会变成其他颜色，有利于我们在写程序时检查所写关键字是否有误。

保存文件很简单，也有很多种方法，这里以最常用的四种来说明。第一种方法是直接单击工具条上的保存图标 ；第二种方法是点击菜单栏的“Feil→Save”；第三种方法是点击菜单栏的“Feil→Save As...”；第四种是按快捷键“Ctrl + S”。

在“文件名(N)”右面的文本框中输入源文件的名字和后缀名时，为了便于管理文件，一般源文件和工程名一致，文件后缀名为 .asm 或 .c，其中 .asm 代表建立的是汇编语言源文件，.c 代表建立的是 C 语言源文件，由于我们是用 C 语言编写程序，所以这里的后缀为 .c，如图 2.10 所示。

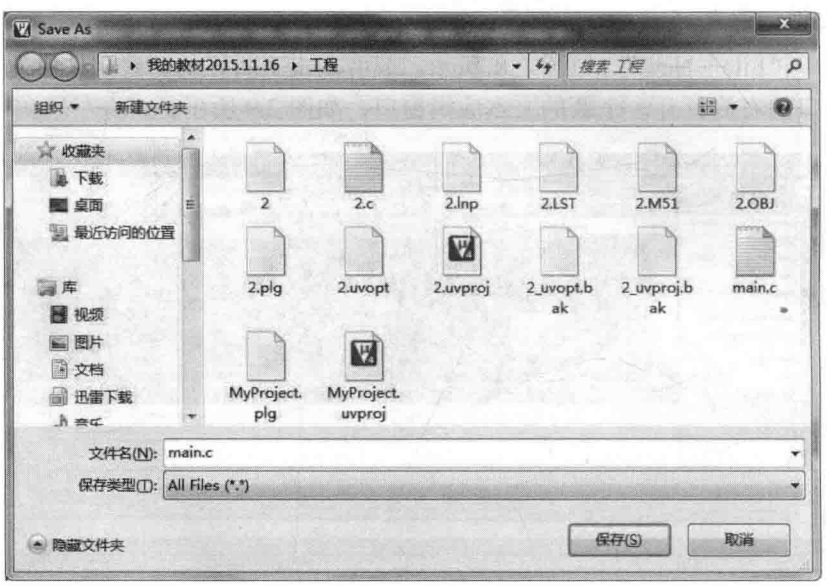

图 2.10　输入源文件后缀名界面

5. 为工程添加源文件

建立好的工程和建立好的程序源文件其实是相互独立的，一个单片机工程是要将源文件和工程联系到一起，这时就需要手动把源程序加入，点击软件界面左上角的“Source Group1”使其反白显示，然后点击鼠标右键，出现一个下拉菜单，选中其中的“Add file to Group‘Source Group1’”如图 2.11 所示。

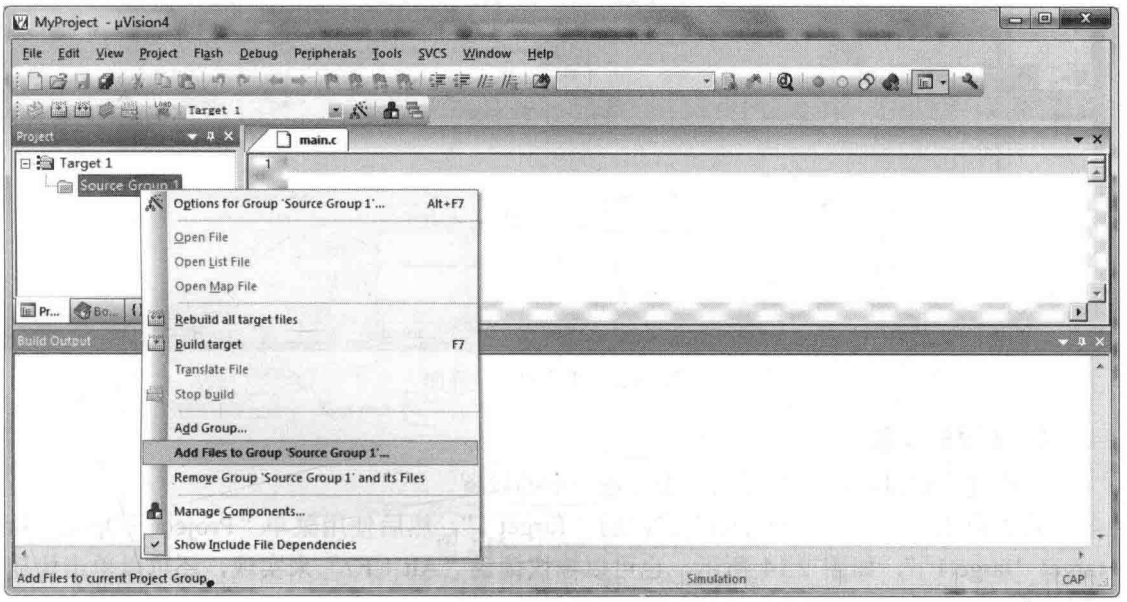

图 2.11　手动加入源程序界面

在执行上面的步骤后会出现一个对话框，要求寻找源文件，需要注意的是，该对话框下面的“文件类型”默认为 C source file(*.c)，也就是以 C 为扩展名的文件。可以找到我们刚刚创建的 main.c 文件，如图 2.12 所示。

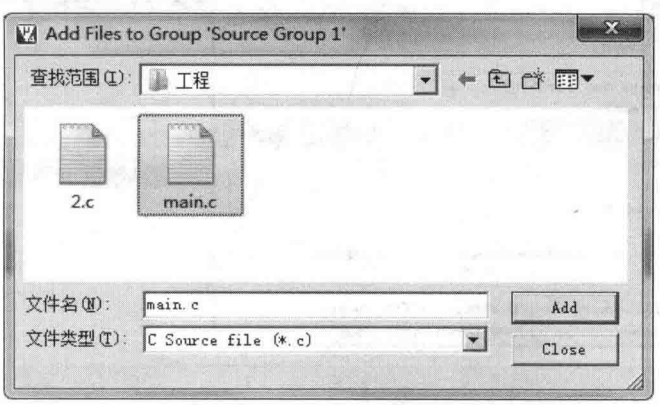

图 2.12　默认源文件界面

点击“Add”，然后点击“Close”即可返回主界面。返回后点击“SourceGroup 1”前面的加号，会发现“main.c”文件已在其中。双击文件名“main.c”，即打开该源程序，如图 2.13 所示。此时就可以在 main.c 源文件上编写 C 语言程序了。

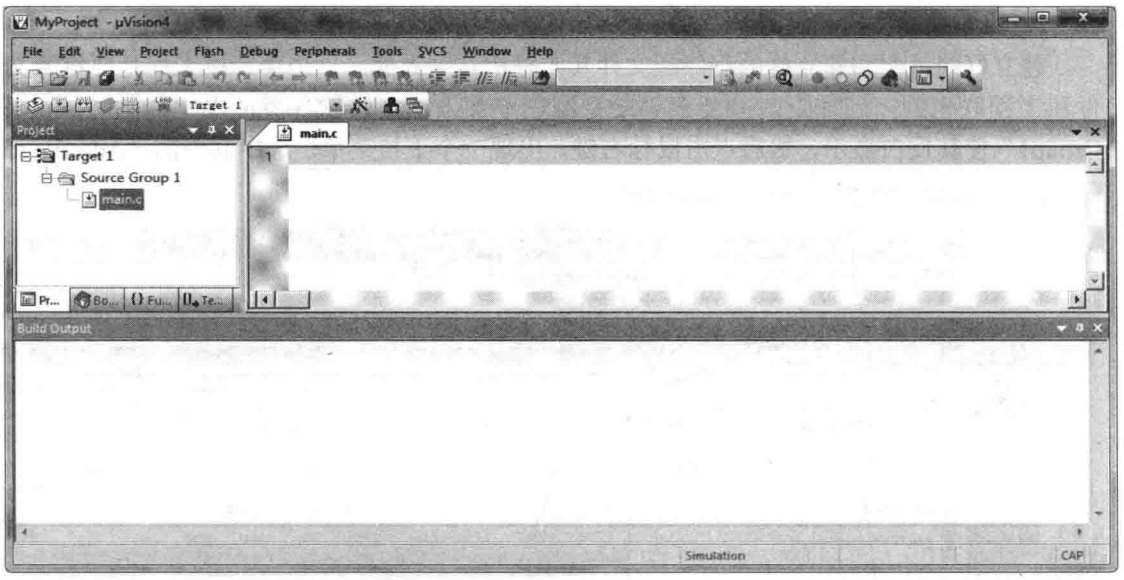

图 2.13　打开源程序界面

6. 工程的设置

工程建立好以后，还要对工程进行进一步的设置。

首先点击左上边的“Project”窗口的“Target 1”，然后使用菜单“Project→Option for target ‘target1’”，如图 2.14 所示，也可以按快捷键“Alt + F7”来实现，还可以单击快捷图标 来完成。

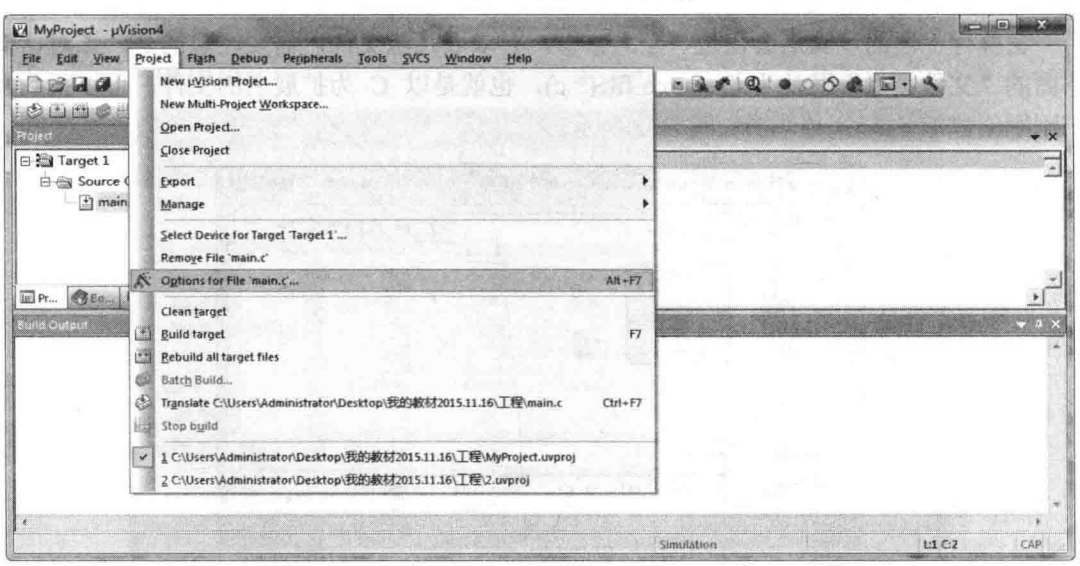

图 2.14　进入工程设置界面

设置对话框中默认的就是 Target 页面，如图 2.15 所示，Xtal 后面的数值是晶振频率值，默认值是所选目标 CPU 的最高可用频率值，对于我们所选的 AT89S52 而言，是 33 MHz，该数值与最终产生的目标代码无关，仅用于软件模拟调试时显示程序执行时间。正确设置

该数值可使显示时间与实际所用时间一致，一般将其设置成与用户的硬件所用晶振频率相同，如果没必要了解程序执行的时间，也可以不设，这里设置为 12.0。

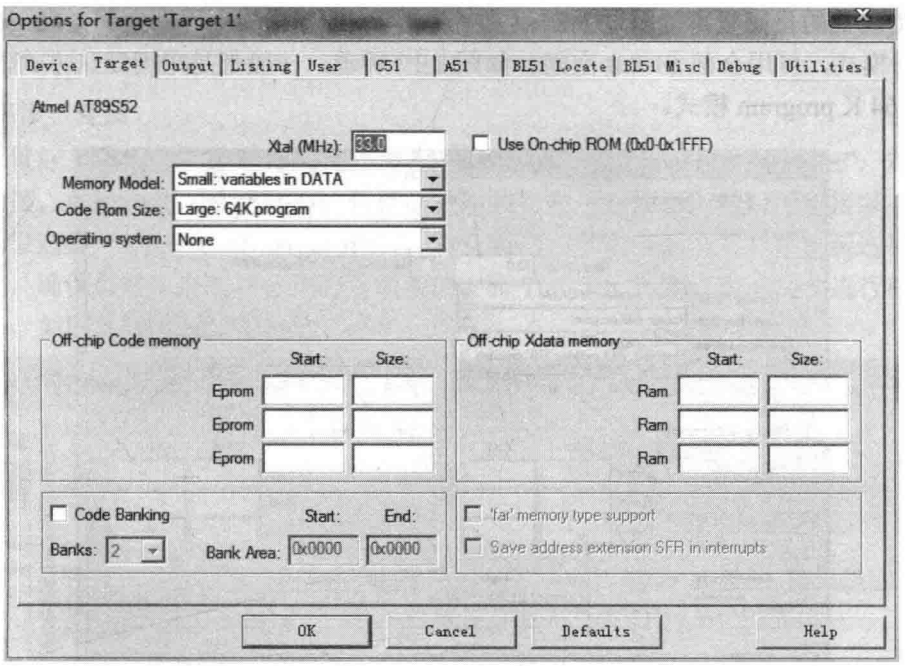

图 2.15　Target 设置界面

Memory Mode 用于设置 RAM 的使用情况，有三个选择项，Small：variables in DATA 是所有变量都在单片机的内部 RAM 中；Compact：variables in PDATA 是可以使用一页外部扩展 RAM；而 Larget：variables in XDATA 则是可以使用全部外部的扩展 RAM，如图 2.16 所示。一般都是采用默认方式，也就是 Small：variables in DATA 方式。

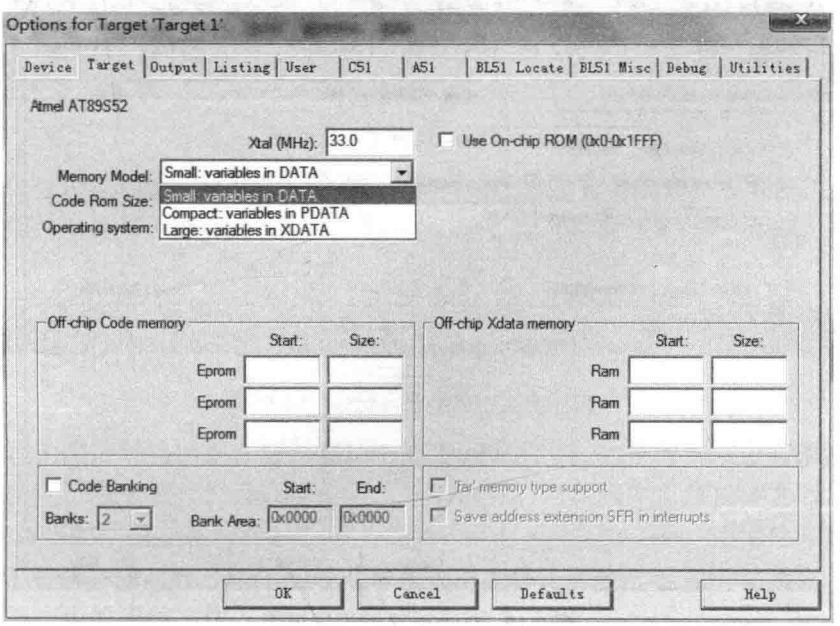

图 2.16　RAM 设置界面