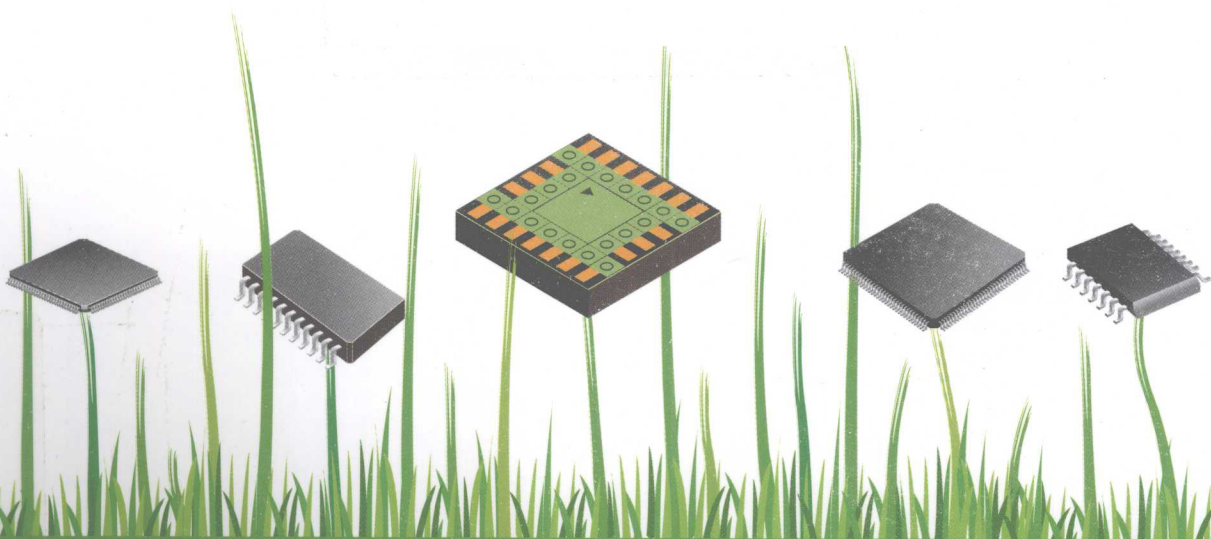


单片机C语言程序 设计实训

100例

——基于AVR+Proteus仿真



彭伟 编著



北京航空航天大学出版社

单片机 C 语言程序设计实训 100 例

——基于 AVR+Proteus 仿真

彭 伟 编著

北京航空航天大学出版社

内 容 简 介

基于 AVR Studio+WinAVR(GCC)组合环境和 Proteus 硬件仿真平台,精心安排了 100 个 AVR 单片机 C 程序设计案例。全书提供了所有案例完整的 C 语言源程序,各案例设计了难易适中的实训目标。

基础设计类案例涵盖 AVR 单片机最基本的端口编程、定时/计数器应用、中断程序设计、A/D 转换、比较器程序设计、EEPROM、Flash、USART 及看门狗程序设计;硬件应用类案例涉及单片机存储器扩展、接口扩展、译码、编码、驱动、光电、机电、传感器、I²C/TWI 及 SPI 接口器件、MMC、红外等器件;综合设计类案例涉及消费类电子产品、仪器仪表及智能控制设备相关技术,相关案例涉及 485 及 RTL8019 的应用。

本书适合用作大专院校学生学习实践 AVR 单片机 C 语言程序设计技术的参考书,也可用作电子工程技术人员、单片机技术爱好者的学习参考书。

图书在版编目(CIP)数据

单片机 C 语言程序设计实训 100 例:基于
AVR+Proteus 仿真/彭伟编著. —北京:北京航空航天大学
大学出版社,2010.5

ISBN 978-7-5124-0068-9

I. ①单… II. ①彭… III. ①单片微型计算机—
C 语言—程序设计 IV. ①TP368.1②TP312

中国版本图书馆 CIP 数据核字(2010)第 068448 号

版权所有,侵权必究。

单片机 C 语言程序设计实训 100 例——基于 AVR+Proteus 仿真

彭 伟 编 著

责任编辑 冯 颖

*

北京航空航天大学出版社出版发行

北京市海淀区学院路 37 号(邮编 100191) <http://www.buaapress.com.cn>

发行部电话:(010)82317024 传真:(010)82328026

读者信箱:bhpress@263.net 邮购电话:(010)82316936

北京时代华都印刷有限公司印装 各地书店经销

*

开本:787×1 092 1/16 印张:36 字数:922 千字

2010 年 5 月第 1 版 2010 年 5 月第 1 次印刷 印数:4 000 册

ISBN 978-7-5124-0068-9 定价:65.00 元

前 言

目前,各高校电类专业都将 C 语言作为专业基础课程纳入教学计划。由于 C 语言功能强大、便于模块化开发、所带库函数非常丰富、编写的程序易于移植,因此,它成为单片机应用系统开发最快速高效的程序设计语言。仅具有 C 语言基础知识但不熟悉单片机指令系统的读者也能很快掌握单片机 C 程序设计技术,C 语言在单片机应用系统设计上的效率优势已经远远高于汇编、BASIC 等开发语言。

单片机 C 程序设计不同于通用计算机应用程序设计,它必须针对具体的微控制器及外围电路来完成。为便于学习单片机应用程序设计和系统开发,很多公司推出了单片机实验箱、仿真器和开发板等,这些硬件设备可用于验证单片机程序、开发和调试单片机应用系统。但由于这些设备价格不菲,它们阻碍了普通读者对单片机技术的学习和研究。令人高兴的是,英国 Labcenter 公司推出了具有单片机系统仿真功能的 Proteus 软件,单片机系统开发通常是基于上位机加目标系统进行的,Proteus 的出现使读者仅用一台 PC 在纯软件环境中完成系统设计与调试成为可能。目前 Proteus 支持 8051、AVR、PIC 等多种单片机,系统库中包含有大量的模拟、数字、光电和机电类元器件,系统还提供了多种虚拟仪器,用 AVR Studio+WinAVR (GCC)开发的程序可以在用 Proteus 设计的仿真电路中调试和交互运行。这无疑为读者学习和提高 AVR 单片机 C 程序设计技术,为单片机应用系统高水平工程师的成长提供了理想平台。

为帮助读者快速提高 AVR 单片机 C 程序设计水平,本书基于 AVR Studio+WinAVR (GCC)组合开发环境和 Labcenter 公司的 Proteus 仿真平台,精心安排了 100 个 AVR 单片机 C 程序设计案例,各案例同时给出了难易适中的实训目标。

前 2 章分别对 AVR - GCC 程序设计和 Proteus 操作基础作了概述。第 3 章基础程序部分给出的案例涵盖 AVR 单片机端口编程、定时/计数器应用、A/D 转换、模拟比较器程序设计、中断程序设计、EEPROM、Flash、USART 及看门狗程序设计,各案例分别对相关知识和技术要点作了阐述与分析,源程序中也给出了丰富的注释信息。第 4 章硬件应用部分针对 AVR 单片机的存储器扩展、接口扩展、译码、编码、驱动、光电、机电、传感器、I²C/TWI 及 SPI 接口器件、MMC、红外等器件给出了数十个案例,对案例中涉及的硬件技术资料亦进行了有针对性的分析,以便于读者快速理解相关代码的编写原理。第 5 章的案例综合应用了单片机内部资源和外部扩展硬件,通过对这些案例的独立分析研究与调试运行,读者用 C 语言开发 AVR 单片机应用系统的能力会得到大幅提升。

本书是单片机 C 语言程序设计实训仿真系列 8051 版之后的第 2 册。为使本书能早日与读者见面,笔者坚持挤出时间不懈耕耘。在编写过程中,刘静、张力、王魏参与了案例的调试与校稿工作,在此对他们深表感谢!本书从选题、撰稿到出版的全过程中,学院领导、学院科研处及高教研究所对本选题始终给予大力支持,并提供项目资助,教务处和信息技术系也一直关注本书的编



写与进展情况,在此一并对学院和部门领导的关心与支持表示由衷感谢!

本书提供完整的案例压缩包,需要的读者可到北京航空航天大学出版社网站 <http://www.buaapress.com.cn> 的“下载中心”免费下载。

由于编者水平有限,加之时间仓促,书中错漏之处在所难免,在此真诚欢迎读者对本书多多提出宝贵意见(笔者的邮箱是:pw95aaa@foxmail.com)。

至此,本套单片机 C 语言程序设计实训仿真系列的 8051 版与 AVR 版已经编写完成,PIC 版正在后续编撰之中,笔者将努力争取使之早日出炉,以飨读者。

彭 伟
2010 年 3 月于武昌

目 录

第 1 章 AVR 单片机 C 语言程序设计概述	1
1.1 AVR 单片机简介	1
1.2 AVR Studio+WinAVR 开发环境安装及应用	4
1.3 AVR-GCC 程序设计基础	7
1.4 程序与数据内存访问	14
1.5 I/O 端口编程	14
1.6 外设相关寄存器及应用	16
1.7 中断服务程序	31
1.8 GCC 在 AVR 单片机应用系统开发中的优势	33
第 2 章 Proteus 操作基础	35
2.1 Proteus 操作界面简介	35
2.2 仿真电路原理图设计	37
2.3 元件选择	39
2.4 仿真运行	44
2.5 Proteus 与 AVR Studio 的联合调试	45
2.6 Proteus 在 AVR 单片机应用系统开发中的优势	46
第 3 章 基础程序设计	48
3.1 闪烁的 LED	48
3.2 左右来回的流水灯	50
3.3 花样流水灯	52
3.4 LED 模拟交通灯	54
3.5 单只数码管循环显示 0~9	57
3.6 8 只数码管滚动显示单个数字	59
3.7 8 只数码管扫描显示多个不同字符	61
3.8 K1~K4 控制 LED 移位	62

3.9	数码管显示 4×4 键盘矩阵按键	65
3.10	数码管显示拨码开关编码	68
3.11	继电器控制照明设备	70
3.12	开关控制报警器	72
3.13	按键发音	74
3.14	INT0 中断计数	76
3.15	INT0 与 INT1 中断计数	79
3.16	TIMER0 控制单只 LED 闪烁	83
3.17	TIMER0 控制流水灯	85
3.18	TIMER0 控制数码管扫描显示	87
3.19	TIMER1 控制交通指示灯	90
3.20	TIMER1 与 TIMER2 控制十字路口秒计时显示屏	94
3.21	用工作于计数方式的 T/C0 实现 100 以内的脉冲或按键计数	98
3.22	用定时器设计的门铃	100
3.23	报警器与旋转灯	103
3.24	100 000 s 以内的计时程序	106
3.25	用 TIMER1 输入捕获功能设计的频率计	109
3.26	用工作于异步模式的 T/C2 控制的可调式数码管电子钟	113
3.27	TIMER1 定时器比较匹配中断控制音阶播放	117
3.28	用 TIMER1 输出比较功能调节频率输出	120
3.29	TIMER1 控制的 PWM 脉宽调制器	123
3.30	数码管显示两路 A/D 转换结果	126
3.31	模拟比较器测试	128
3.32	EEPROM 读/写与数码管显示	130
3.33	Flash 程序空间中的数据访问	136
3.34	单片机与 PC 机双向串口通信仿真	141
3.35	看门狗应用	147
第 4 章	硬件应用	150
4.1	74HC138 与 74HC154 译码器应用	150
4.2	74HC595 串入并出芯片应用	153
4.3	用 74LS148 与 74LS21 扩展中断	157
4.4	62256 扩展内存实验	160
4.5	用 8255 实现接口扩展	163
4.6	可编程接口芯片 8155 应用	168
4.7	可编程外围定时/计数器 8253 应用	173
4.8	数码管 BCD 解码驱动器 7447 与 4511 应用	178
4.9	8×8 LED 点阵屏显示数字	181
4.10	8 位数码管段位复用串行驱动芯片 MAX6951 应用	183

4.11	串行共阴显示驱动器 MAX7219 与 7221 应用	188
4.12	16 段数码管演示	193
4.13	16 键解码芯片 74C922 应用	196
4.14	1602 LCD 字符液晶测试程序	199
4.15	1602 液晶显示 DS1302 实时时钟	205
4.16	1602 液晶工作于 4 位模式实时显示当前时间	211
4.17	2×20 串行字符液晶演示	214
4.18	LGM12864 液晶显示程序	217
4.19	PG160128A 液晶图文演示	226
4.20	TG126410 液晶串行模式显示	247
4.21	用带 SPI 接口的 MCP23S17 扩展 16 位通用 I/O 端口	257
4.22	用 TWI 接口控制 MAX6953 驱动 4 片 5×7 点阵显示器	262
4.23	用 TWI 接口控制 MAX6955 驱动 16 段数码管显示	266
4.24	用 DAC0832 生成多种波形	270
4.25	用带 SPI 接口的数/模转换芯片 MAX515 调节 LED 亮度	273
4.26	正反转可控的直流电机	276
4.27	正反转可控的步进电机	279
4.28	DS18B20 温度传感器测试	282
4.29	SPI 接口温度传感器 TC72 应用测试	293
4.30	SHT75 温、湿度传感器测试	299
4.31	用 SPI 接口读/写 AT25F1024	309
4.32	用 TWI 接口读/写 24C04	318
4.33	MPX4250 压力传感器测试	326
4.34	MMC 存储卡测试	329
4.35	红外遥控发射与解码仿真	340
第 5 章	综合设计	348
5.1	多首电子音乐的选播	348
5.2	电子琴仿真	353
5.3	普通电话机拨号键盘应用	357
5.4	1602 LCD 显示仿手机键盘按键字符	363
5.5	数码管模拟显示乘法口诀	369
5.6	用 DS1302 与数码管设计的可调电子钟	372
5.7	用 DS1302 与 LGM12864 设计的可调式中文电子日历	380
5.8	用 PG12864LCD 设计的指针式电子钟	393
5.9	高仿真数码管电子钟	401
5.10	1602 LCD 显示的秒表	409
5.11	用 DS18B20 与 MAX6951 驱动数码管设计的温度报警器	413
5.12	用 1602 LCD 与 DS18B20 设计的温度报警器	421



5.13	温控电机在 L298 驱动下改变速度与方向运行	431
5.14	PG160128 中文显示日期时间及带刻度显示当前温度	439
5.15	液晶屏曲线显示两路 A/D 转换结果	447
5.16	用 74LS595 与 74LS154 设计的 16×16 点阵屏	452
5.17	用 8255 与 74LS154 设计的 16×16 点阵屏	457
5.18	8×8 LED 点阵屏仿电梯数字滚动显示	461
5.19	用内置 EEPROM 与 1602 液晶设计的带 MD5 加密的电子密码锁	466
5.20	12864LCD 显示 24C08 保存的开机画面	480
5.21	12864LCD 显示 EPROM27C256 保存的开机画面	488
5.22	I ² C - AT24C1024×2 硬字库应用	491
5.23	SPI-AT25F2048 硬件字库应用	498
5.24	带液晶显示的红外遥控调速仿真	505
5.25	能接收串口信息的带中英文硬字库的 80×16 点阵显示屏	511
5.26	用 AVR 与 1601 LCD 设计的计算器	523
5.27	电子秤仿真设计	531
5.28	模拟射击训练游戏	537
5.29	PC 机通过 485 远程控制单片机	546
5.30	用 IE 访问 AVR+RTL8019 设计的以太网应用系统	550
参考文献		568

第 1 章

AVR 单片机 C 语言程序设计概述

1997 年,美国 Atmel 公司将其先进的 Flash 技术与 8051 单片机结合起来,推出了 8 位 AVR 单片机。与传统的采用复杂指令系统(CISC)的 8051 单片机不同的是,AVR 单片机采用了更适合中高档电子产品和嵌入式系统应用需求的精简指令系统(RISC)。

为适应不同层次与不同场合的应用需要,Atmel 公司推出了多个系列的 AVR 单片机,它们广泛应用于工控系统、计算机外部设备、通信设备、仪器仪表及应用类电子产品。本书所有案例使用的是 ATmega 系列单片机,涉及的型号有 ATmega8515/8535、ATmega8/16/32/64/128。

作为 Atmel 公司免费推出的 AVR 单片机开发平台,AVR Studio 提供了编写和调试 AVR 单片机应用程序的集成开发环境。它为源程序编写、设备编程、仿真及片上调试提供一个项目管理工具、源代码编辑器、模拟器、集成环境和前端。AVR Studio 支持全部的 Atmel AVR 工具集,每个版本总是包含有最新的 AVR 器件和工具支持。

但是,当前版本的 AVR Studio 仅提供汇编语言编译器,并未提供 C/C++ 程序编译器。要在 AVR Studio 集成环境中开发 AVR 单片机 C 程序,显然还需要安装配置第三方提供的 C/C++ 语言程序编译器。本书案例开发所使用的 WinAVR 是一套用于 Atmel AVR 系列 RISC 微处理器程序开发的开源的软件开发工具集,它以著名的自由软件 GCC 为 C/C++ 编译器。

使用 AVR Studio 提供的程序开发与调试前端及 WinAVR 提供的 AVR-GCC 编译程序组合搭建的 AVR 单片机 C 语言程序开发平台,可大大降低开发成本,缩短开发周期,大幅提高开发效率,程序可读性好且易于移植。

本书的编写,旨在进一步提高读者的 AVR 单片机 C 语言程序开发能力,全书的 100 个案例全部在 AVR Studio+WinAVR 环境下编写并调试通过,同时给出了完整 Proteus 仿真电路。当前版本的 AVR Studio 已经支持内嵌 Proteus 进行跟踪调试与仿真。

阅读使用本书之前要求读者已经学习了基本的 AVR 单片机 C 程序设计技术,本章仅介绍使用 C 语言设计单片机应用系统必须参考和重点掌握的技术内容,这些内容会对阅读、调试、研究本书案例及进行设计实训提供重要参考。

1.1 AVR 单片机简介

本书案例使用的 AVR 单片机有 ATmega8515/8535、ATmega8/16/32/64/128,图 1-1 给出了全书案例中出现最多的 ATmega16(L)单片机的不同封装形式及引脚分布,本节仅对该单片机端口及引脚作简要介绍。

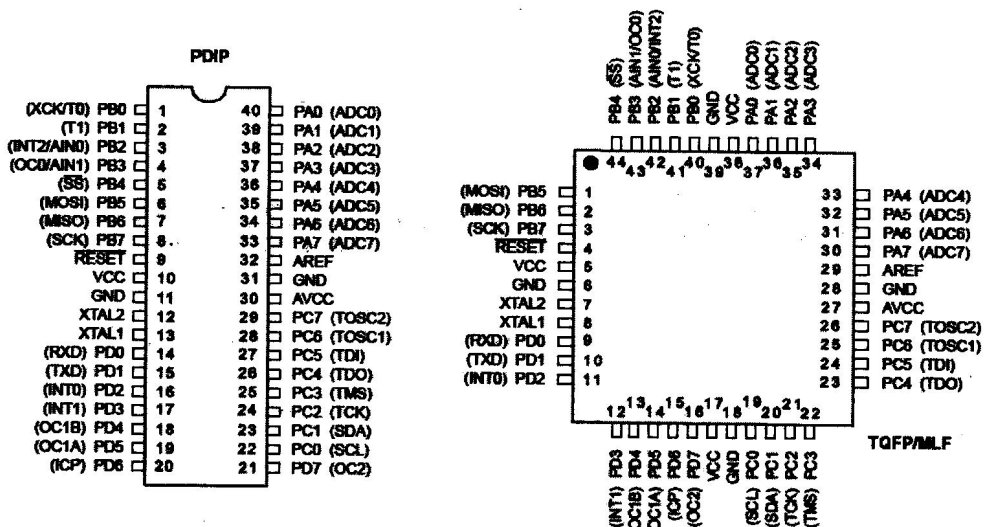


图 1-1 ATmega16(L)单片机不同封装形式及引脚

ATmega16(L)的主要特征如下:

(1)非易失性程序与数据存储(Nonvolatile Program and Data Memories)

- 16 KB 的系统内可编程 Flash,可擦写 10000 次,支持 ISP(在系统编程)和 IAP(在应用编程)。
- 单片机内部数据存储 SRAM 空间达 1 KB。
- 512 字节的 EEPROM 可擦写 100 000 次,可以在系统掉电时保存用户数据信息。

(2)端口及外设特征(Peripheral Features)

- 32 个可编程 I/O 口分为 PA~PD 共 4 组,每组 8 位。
- 2 个具有独立预分频器和比较器功能的 8 位定时/计数器,1 个具有预分频器、比较功能和捕获功能的 16 位定时/计数器。
- 具有独立振荡器的实时计数器 RTC。
- 4 通道 PWM。
- 内置 8 通道 10 位精度的逐次逼近式 A/D 转换器(ADC),支持单端和双端差分信号输入,内含增益可编程运算放大器。
- 片内模拟比较器。
- 面向字节的两线式串行接口 TWI(Two-Wire serial Interface)。
- 可工作于主机/从机模式的串行接口 SPI (Serial Peripheral Interface)。
- 2 个可编程的串行 USART。
- 具有独立片内振荡器的可编程看门狗定时器。

(3)工作电压与速度等级

- ATmega16L:2.7 ~ 5.5 V;ATmega16:4.5 ~ 5.5 V。
- ATmega16L:0 ~ 8 MHz;ATmega16:0 ~ 16 MHz。

在了解 ATmega16(L) 单片机的主要特征以后,再来看一下单片机的引脚说明:

(1) 4 个通用 I/O 端口

PA~PD 端口都是 8 位的双向 I/O 端口,具有可编程的内部上拉电阻,其输出缓冲器具有对称的驱动特性,可以输出和吸收较大电流。作为输入端口使用时,如果使能内部上拉电阻,端口被外部电路拉低时将输出电流。

(2) PA~PD 端口的第二功能

PA 端口(PA7~PA0)可作为 A/D 转换器的模拟输入端 ADC7~ADC0。

PB 端口(PB7~PB0)的第二功能如下:

PB7: SCK (SPI 总线的串行时钟);

PB6: MISO (SPI 总线的主机输入/从机输出信号);

PB5: MOSI (SPI 总线的主机输出/从机输入信号);

PB4: SS (SPI 从机选择引脚);

PB3: AIN1 (模拟比较负输入)/OC0 (T/C0 输出比较匹配输出);

PB2: AIN0 (模拟比较正输入)/INT2 (外部中断 2 输入);

PB1: T1 (T/C1 外部计数器输入);

PB0: T0 (T/C0 外部计数器输入)/XCK (USART 外部时钟输入/输出)。

PC 端口(PC7~PC0)引脚第二功能如下:

PC7: TOSC2 (定时振荡器引脚 2);

PC6: TOSC1 (定时振荡器引脚 1);

PC5: TDI (JTAG 测试数据输入);

PC4: TDO (JTAG 测试数据输出);

PC3: TMS (JTAG 测试模式选择);

PC2: TCK (JTAG 测试时钟);

PC1: SDA (两线串行总线数据输入/输出线);

PC0: SCL (两线串行总线时钟线)。

PD 端口(PD7~PD0)的第二功能如下:

PD7: OC2 (T/C2 输出比较匹配输出);

PD6: ICP1 (T/C1 输入捕获引脚);

PD5: OC1A (T/C1 输出比较 A 匹配输出);

PD4: OC1B (T/C1 输出比较 B 匹配输出);

PD3: INT1 (外部中断 1 输入引脚);

PD2: INT0 (外部中断 0 输入引脚);

PD1: TXD (USART 输出引脚);

PD0: RXD (USART 输入引脚)。

(3) 其他引脚

RESET: 复位输入引脚,持续时间超过最小门限时间的低电平将引起系统复位。

XTAL1: 反向振荡放大器与片内时钟操作电路的输入端。

XTAL2: 反向振荡放大器的输出端。

AVCC: 端口 A 与 A/D 转换器的电源,不使用 ADC 时,该引脚应直接与 VCC 连接,使用



ADC 时应通过一个低通滤波器与 VCC 连接。

AREF: A/D 的模拟基准输入引脚。

VCC/GND: 数字电路的电源/地。

Proteus 仿真时电源已默认连接, 仿真电路图中电源连接全部默认。AVCC 在未使用 ADC 的多数电路中没有连接 VCC, 在实物电路设计时应注意连接 VCC。

1.2 AVR Studio+WinAVR 开发环境安装及应用

AVR Studio 是 Atmel 官方针对 AVR 系列单片机推出的集成开发环境, 它集开发调试于一体, 有很好的用户界面与很好的稳定性。由于 AVR Studio 仅支持编译汇编语言程序, 不支持对 C 语言程序的编译, 要基于 AVR Studio 搭建 AVR 单片机 C 语言程序开发环境, 除免费下载安装 AVR Studio 以外, 还需要下载安装 C/C++ 编译器。本书通过下载安装 WinAVR 来提供 AVR-GCC 编译器。

搭建基于 AVR Studio+WinAVR 的案例开发环境时, 所使用的安装包分别为 WinAVR-20090313-install.exe 及 AvrStudio416Setup.exe, 这两个安装包都可以从网上免费下载。在实际应用中, 可根据需要随时获取最新版本的安装包。

当前版本的 AVR Studio 对中文路径支持得还不够好, 在创建新项目时建议用英文或数字等非全角字符命名文件夹或文件。图 1-2 所示窗口中创建的 LCD_Test 项目保存于 E:\my_avr_c 文件夹下, 单击 Next 按钮后可进一步选择设备(Device)。

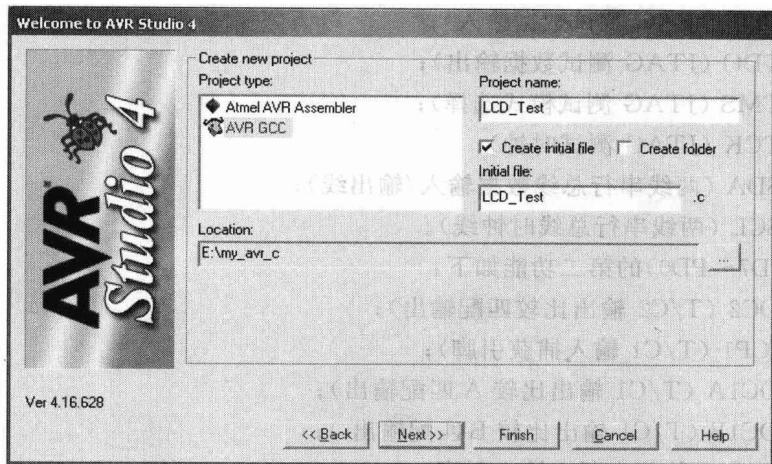


图 1-2 创建新项目

图 1-3 所示为 AVR Studio 主窗体, 左边的 AVR GCC 窗口以树形方式列出了当前项目的所有相关文件, 该窗口中除初始时创建的 LCD_Test.c 文件以外, 还添加了 LCD1602.c。编写完该项目 C 程序后, 在开始编译并生成 HEX、ELF、EEP 等文件之前, 要先单击工具栏上的编辑当前配置选项(Edit Current Configuration Options)按钮。该按钮在图 1-3 所示窗口第 2 条工具栏的最右边。

图 1-4 是当前项目选项设置窗口, 在常规(General)选项中可以看到当前项目的默认输出文件夹是 default。如果还没有设置当前 AVR 单片机选型, 可在设备(Device)下拉框中选

取,当前选择的是 ATmega16。设备下面是时钟频率(Frequency)设置文本框,当前设置的频率是 4 MHz。编译优化(Optimization)选项当前选择是“-Os”,它优化所输出的可执行程序文件的大小(Optimize for size)。有关该窗口中配置选项的更多细节,可单击帮助按钮查看。

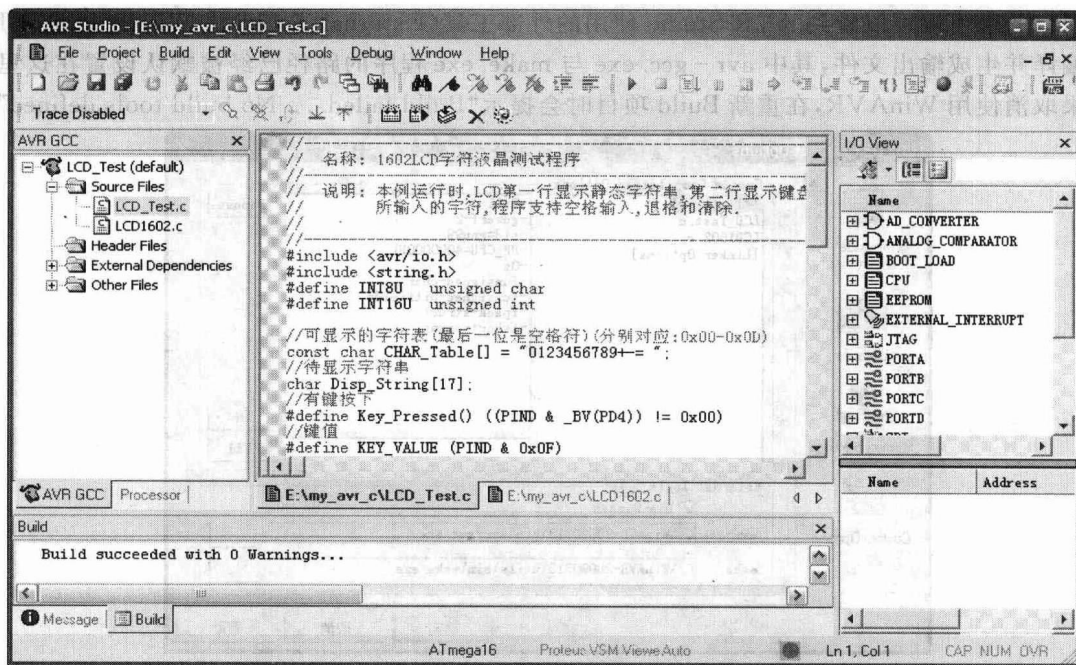


图 1-3 AVR Studio 主窗体

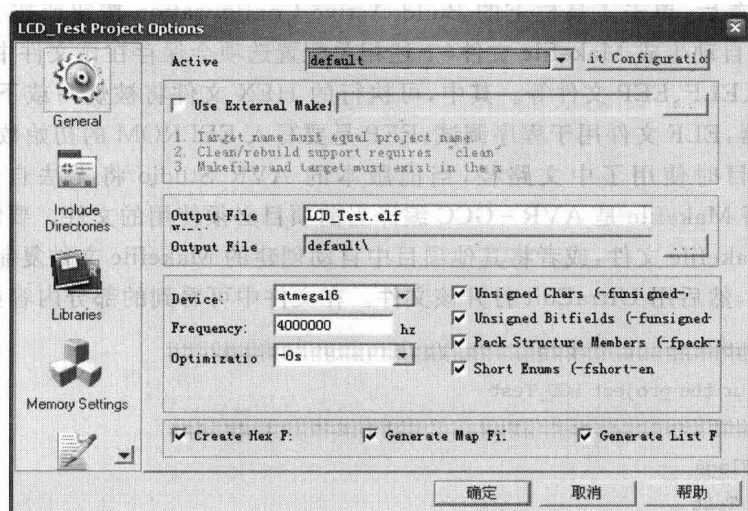


图 1-4 当前项目的常规配置选项窗口

该窗口的 Unsigned Chars(-funsigned char)设置选项应引起重视,此选项是默认选中的,它将程序中的 char 类型默认为无符号类型,这一点与 Keil C 是不一样的。假如程序中出现 -50~125℃ 范围内的正负温度比较,编写程序时就需要使用 signed char 类型定义变量,



而不能直接使用 char 类型进行定义,除非取消该设置。

该窗口中的其他配置选项可保持为默认值,不作任何改动。

当打开项目选项中的自定义选项(Custom Options)时可看到图 1-5 所示的配置窗口,在该窗口的最下端可以看到 AVR Studio 使用的外部工具(External Tools)是 WinAVR,它被用来编译并生成输出文件,其中 avr-gcc.exe 与 make.exe 程序的路径已经被默认设置在这里。如果取消使用 WinAVR,在重新 Build 项目时会提示“Build failed... No build tools defined”。

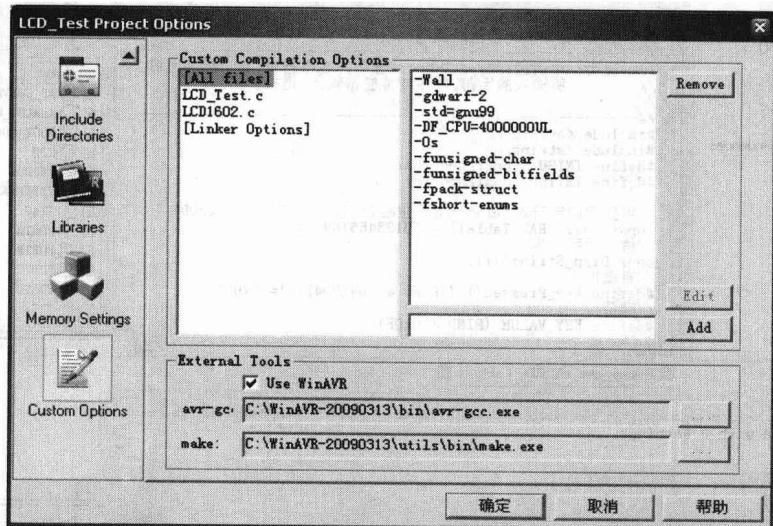


图 1-5 自定义选项配置窗口

完成所有设置后,单击工具栏上的 Build Active Configuration 按钮或按下 F7,这时 default 文件夹下会自动生成 Makefile 文件(上述相关配置选项会保存在该文件中),并生成输出文件,包括 HEX、ELF、EEP 文件等。其中,可执行的 HEX 文件将被烧写或下载到单片机的 Flash 程序存储器,ELF 文件用于程序调试,EFP 是要写入 EEPROM 的初始数据文件。

如果创建项目时使用了中文路径,当前版本的 AVR Studio 将无法自动创建或修改 Makefile 文件,而 Makefile 是 AVR-GCC 编译当前项目必须使用的文件。要解决这一问题,可以手动创建 Makefile 文件,或者将其他项目中自动创建的 Makefile 文件复制到当前项目的 default 文件夹下,然后用 UltraEdit 打开该文件。在文件中可看到的部分内容如下:

```
#####
# Makefile for the project LCD_Test
#####
## General Flags
PROJECT = LCD_Test
MCU = atmega16
TARGET = LCD_Test.elf
CC = avr-gcc
.....
CFLAGS + = ..... -DF_CPU = 4000000UL -Os -funsigned-char .....
.....
```

```

## Objects that must be built in order to link
OBJECTS = LCD_Test.o LCD1602.o
.....
## Compile
LCD_Test.o: ../LCD_Test.c
    $(CC) $(INCLUDES) $(CFLAGS) -c $<
LCD1602.o: ../LCD1602.c
    $(CC) $(INCLUDES) $(CFLAGS) -c $<
.....

```

该文件中的以下相关项目要在 UltraEdit 内根据需要进行修改并保存：

PROJECT——设置输出的项目名称，这需要根据当前项目名称进行修改；

MCU——设置当前项目所选用微控制器；

TARGET——设置输出目标调试文件 (*.ELF)；

DF_CPU——设置所选择的时钟频率；

OBJECTS——列出 build 时所使用的目标文件 (*.o)，有多个目标文件时中间用空格隔开；

Compile——列出了当前项目中的所有源程序文件名 (*.c) 及生成的目标文件名 (*.o)，根据当前新建的项目文件中的源程序文件名称及文件个数，这里需要进行相应的增删或修改。

通过 UltraEdit 修改 Makefile 以后，处于中文路径下的 AVR 单片机 GCC 程序项目仍然可以正常编译并生成输出文件。

有关在 AVR Studio 中通过 Debug 菜单选择设备及调试平台 (Select Device and Debug Platform)，对当前生成的程序文件进行跟踪调试的方法将在第 2 章介绍。

1.3 AVR - GCC 程序设计基础

AVR - GCC 是一款优秀的 AVR 编译软件，是 GUN C 编译器在 AVR 上的移植，支持多种操作系统。本节不准备全面讲述 AVR - GCC 程序设计的所有基础内容，下面仅列举部分编写调试本书案例程序时需要引起注意的部分和在编写过程中容易出现错误的部分。本节最后还列出了部分 AVR - GCC 对标准 C 语言的扩展功能。

1. 基本数据类型、有符号与无符号数应用、数位分解、位操作

在讨论本节内容之前需要先熟悉 GCC 基本数据类型，本书 C 程序中所使用的部分基本数据类型如表 1-1 所列。表中的第 2 列与第 3 列分别是 GCC 头文件 <stdint.h> 及本书所使用的各种数据类型的重定义。要使用精确定义的数据宽度，建议引入头文件 <stdint.h>，使用该文件所定义的类型，这些类型都以“_t”结尾。另外，如果要在程序中使用布尔类型 (bool，取值为 true/false)，可在程序中引入头文件 <stdbool.h>。



表 1-1 AVR-GCC 部分常用基本数据类型

数据类型	stdint. h 定义	本书定义	长度/位	取值范围
signed char (char)	int8_t	INT8	8	-128~127
unsigned char	uint8_t	INT8U	8	0~255
char			8	取值范围为上述两者之一,具体范围由配置选项决定
int (signed int)	int16_t	INT16	16	-32768~32767
unsigned int	uint16_t	INT16U	16	0~65535
long (signed long)	int32_t	INT32	32	-2147483648~2147483647
unsigned long	uint32_t	INT32U	32	0~4294967295
float/double			32	$-3.4 \times 10^{38} \sim 3.4 \times 10^{38}$

本书大多数案例使用的都是无符号数,对于 0~255 以内的整数,本书全部定义为 INT8U 类型,相当于字节类型 BYTE 或<stdint. h>定义的 uint8_t 类型;对于 0~65535 以内的整数,本书定义为 INT16U 类型,相当于字类型 WORD 或<stdint. h>定义的 uint16_t 类型。另外,GCC 不支持 float 类型,它将 float 直接解释为 double 类型。

如果涉及正负数的处理,在定义类型时要注意使用 signed 类型。例如温度控制程序中有正负温度,传感器实际上可处理的温度范围为-55~125℃。为使程序能对温度值进行正确比较,程序中将温度类型定义为 INT8 类型(即 signed char 或 int8_t 类型,其取值范围为-128~127)。在 Keil C 中 char 类型默认为 signed char,但在 AVR Studio 的配置窗口中,char 类型默认为 unsigned char 类型,在定义可能出现正负值的温度变量时,不能定义使用 char temp,除非修改配置选项,将 char 默认为 signed char。

本书大量案例要将整数或浮点数显示在数码管上,这需要对待显示数据各数位进行分解,例如:

```
INT8U d = 227;
INT8U c[3];
c[0] = d/100;
c[1] = d/10 % 10;
c[2] = d % 10;
```

又如:

```
float x = 123.45;
```

如果要得到 x 的各个数位,可以先将 x 乘以 100,然后再分解各数位:

```
INT16U y = x * 100;
INT8U c[5], i;
for (i = 4; i != 0xFF; i--)
{
    c[i] = y % 10; y /= 10;
}
```