

# $\mu$ Vision2单片机 应用程序开发指南

尹 勇 李 宇 编著



科学出版社  
[www.sciencep.com](http://www.sciencep.com)

# **$\mu$ Vision2 单片机应用程序开发指南**

尹 勇 李 宇 编著

科 学 出 版 社

北 京

## 内 容 简 介

本书重点阐述在 Windows 集成开发环境  $\mu$ Vision2 下,使用 C51 高级语言开发和调试 C 语言程序,帮助读者达到熟练掌握、使用  $\mu$ Vision2。书中所有的例子都经过认真审核,以确保无技术问题。

本书可作为从事嵌入式应用系统和单片机开发的技术人员的参考书,也可供高等学校工科电子类专业师生参考。

### 图书在版编目 (CIP) 数据

$\mu$ Vision2 单片机应用程序开发指南/尹勇,李宇编著.—北京:科学出版社,2004

ISBN 7-03-014663-8

I.  $\mu$ … II. ①尹… ②李… III. 单片微型计算机-程序设计-指南 IV.TP368.1-62

中国版本图书馆 CIP 数据核字 (2004) 第 124024 号

责任编辑:吕建忠 韩 洁/责任校对:耿 耘

责任印制:吕春珉/封面设计:飞天创意

科学出版社出版

北京东黄城根北街 16 号

邮政编码:100717

<http://www.sciencep.com>

世界知识印刷厂印刷

科学出版社发行 各地新华书店经销

\*

2005 年 2 月第 一 版 开本:787×1092 1/16

2005 年 2 月第一次印刷 印张:24

印数:1—4 000 字数:554 000

定价:35.00 元

(如有印装质量问题,我社负责调换(世知))

## 前 言

8051 系列微处理器被广泛应用于类产品中,很多制造商都提供 8051 系列单片机,例如 Intel、Philips 和 Siemens 等。这些制造商给 51 系列单片机加入了大量新的性能和外部功能,例如 I2C 总线接口、模拟量和数字量的转换、看门狗、PWM 输出等。这些功能使得 8051 单片机很适合作为厂家产品的基本构架,方便进行各种开发应用。

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。与汇编语言相比,C 语言在功能、结构性、可读性、可维护性上有明显的优势,并且易学易用。Keil C51 提供丰富的库函数和功能强大的集成开发调试工具,全 Windows 界面,生成的目标代码效率非常高,多数语句生成的汇编代码很紧凑,且容易理解,在开发大型软件时更能体现高级语言的优势。

单片机系统的开发和应用非常广泛,传统的单片机方面的书籍主要介绍内部结构、工作原理、指令系统和硬、软件开发,而 C 语言的书籍主要介绍语法和程序设计方面的知识。本书不是介绍 Keil C51 语言和单片机硬件方面的知识,而是指导读者如何使用  $\mu$ Vision2 开发和调试单片机的 C 语言程序。本书采用实例方式进行阐述,循序渐进,简洁明了。读者只要按照实例步骤去实践,就能在最短的时间内达到使用  $\mu$ Vision2 开发和调试程序的中、高级能力。

本书的章节安排如下:

第 1 章 Keil C51 的简单回顾,让读者在最短的时间内具备单片机的 C 语言编程能力。

第 2 章 Keil C51 开发工具简介,介绍单片机 C 语言开发工具的发展歷程和各种开发工具的特点。

第 3 章  $\mu$ Vision2 的集成开发环境,详细介绍 Keil C51 的 Windows 集成开发环境  $\mu$ Vision2 的菜单命令、工具、窗口及其使用。

第 4 章 用  $\mu$ Vision2 建立项目,介绍项目程序的建立、设置和编译等,是本书的重点章节。

第 5 章 用  $\mu$ Vision2 调试项目,介绍如何用 Keil C51 的 Windows 集成开发环境  $\mu$ Vision2 调试项目程序,包括调试工具的使用、调试命令和函数的建立以及各种调试技巧,是本书的重点章节。

第 6 章  $\mu$ Vision2 的实时操作系统 RTX-51,介绍 8051 单片机系列的多任务处理内核程序,并给出了一个详细的实例。

第 7 章 在  $\mu$ Vision2 中使用片上外设,介绍包括特殊功能寄存器、中断服务寄存器、并行 I/O 端口、串行接口、定时器/计数器和数/模转换器等外设模块的使用。

第 8 章 使用  $\mu$ Vision2 内嵌的 Monitor-51,介绍  $\mu$ Vision2 内嵌的目标系统调试、模块 Monitor-51 的配置、使用以及疑难解答,并给出一个实例。

本书由尹勇和李宇执笔，参与本书编写的还有关荣锋、李洪杰、范良志、张超勇和朱传军。本书在编写过程中受到连燕波和柯艳青的鼎力帮助，特表感谢。

由于作者水平有限，难免出现错误和不妥之处，恳请广大读者批评指正。

作 者

2004年2月23日

于华中科技大学

# 目 录

|                                       |    |
|---------------------------------------|----|
| 第 1 章 Keil C51 的简单回顾 .....            | 1  |
| 1.1 51 单片机的特点 .....                   | 1  |
| 1.1.1 AT89C51 的引脚功能 .....             | 2  |
| 1.1.2 AT89C2051 简介 .....              | 4  |
| 1.2 C51 程序的基本结构 .....                 | 5  |
| 1.3 C51 的标识符与关键字 .....                | 7  |
| 1.4 C51 的数据类型 .....                   | 9  |
| 1.5 C51 的常量和变量 .....                  | 13 |
| 1.5.1 C51 的常量 .....                   | 13 |
| 1.5.2 C51 的变量 .....                   | 15 |
| 1.6 C51 的函数 .....                     | 18 |
| 1.6.1 函数的说明 .....                     | 18 |
| 1.6.2 函数的定义 .....                     | 19 |
| 1.6.3 函数的调用 .....                     | 19 |
| 1.7 C51 的数组与指针 .....                  | 20 |
| 1.7.1 C51 的数组 .....                   | 20 |
| 1.7.2 C51 的指针 .....                   | 21 |
| 1.8 C51 的结构与联合 .....                  | 24 |
| 1.8.1 C51 的结构 .....                   | 24 |
| 1.8.2 C51 的联合 .....                   | 27 |
| 1.9 C51 类型定义 .....                    | 29 |
| 1.10 C51 的编译预处理 .....                 | 29 |
| 1.10.1 宏定义 .....                      | 30 |
| 1.10.2 文件包含 .....                     | 33 |
| 1.10.3 条件编译 .....                     | 34 |
| 第 2 章 Keil C51 开发工具简介 .....           | 37 |
| 2.1 单片机应用系统设计的基本步骤 .....              | 37 |
| 2.2 DOS 下的 C51 开发工具 .....             | 39 |
| 2.2.1 C51 开发工具介绍 .....                | 39 |
| 2.2.2 Keil C51 的 C 编译器 .....          | 40 |
| 2.2.3 Keil C51 的 A51 宏汇编器 .....       | 41 |
| 2.2.4 Keil C51 的 BL51 代码连接器/定位器 ..... | 42 |
| 2.2.5 Keil C51 的 OC51 目标文件转换器 .....   | 43 |
| 2.2.6 Keil C51 的 OH51 目标十六进制转换器 ..... | 44 |

|                                   |     |
|-----------------------------------|-----|
| 2.2.7 Keil C51 的 LIB51 库文件管理器     | 44  |
| 2.3 Windows 下的 C51 开发工具           | 44  |
| 2.3.1 $\mu$ Vision1 版本            | 45  |
| 2.3.2 $\mu$ Vision2 版本            | 47  |
| 2.4 $\mu$ Vision2 的安装             | 51  |
| 2.4.1 系统需求                        | 51  |
| 2.4.2 安装步骤及注意事项                   | 51  |
| 2.4.3 $\mu$ Vision2 的具体安装过程       | 51  |
| 2.5 $\mu$ Vision2 安装后的文件组织结构      | 54  |
| 2.6 一个完整的应用实例                     | 54  |
| 第3章 $\mu$ Vision2 的集成开发环境         | 59  |
| 3.1 $\mu$ Vision2 项目管理窗口          | 59  |
| 3.1.1 目标、文件组和文件的管理                | 59  |
| 3.1.2 项目窗口中的文件和文件组的属性             | 62  |
| 3.2 $\mu$ Vision2 的菜单栏            | 64  |
| 3.3 $\mu$ Vision2 工具栏的使用          | 76  |
| 3.4 $\mu$ Vision2 快捷键的使用          | 79  |
| 3.5 $\mu$ Vision2 的各种窗口           | 81  |
| 3.5.1 设置窗口属性                      | 81  |
| 3.5.2 源代码编辑窗口                     | 84  |
| 3.5.3 反汇编窗口                       | 85  |
| 3.5.4 Watch & Call Stack 窗口       | 86  |
| 3.5.5 Memory 窗口                   | 87  |
| 3.5.6 CPU 寄存器窗口                   | 89  |
| 3.5.7 串行窗口                        | 89  |
| 3.5.8 性能分析窗口                      | 90  |
| 3.5.9 代码覆盖窗口                      | 91  |
| 3.5.10 符号观察窗口                     | 92  |
| 第4章 用 $\mu$ Vision2 建立项目          | 95  |
| 4.1 启动 $\mu$ Vision2 并创建一个项目      | 95  |
| 4.1.1 创建一个新的项目                    | 95  |
| 4.1.2 新建一个源文件                     | 96  |
| 4.2 增加和配置启动代码                     | 97  |
| 4.3 $\mu$ Vision2 的 CPU 和程序启动代码详解 | 98  |
| 4.4 为目标设置工具选项                     | 101 |
| 4.4.1 配置对话框介绍                     | 101 |
| 4.4.2 例子项目的设置                     | 102 |
| 4.5 编译项目并生成 HEX 文件                | 123 |
| 4.6 代码分块                          | 124 |

|        |                                           |     |
|--------|-------------------------------------------|-----|
| 4.7    | 使用资源浏览器 .....                             | 138 |
| 4.8    | Keil C51 与汇编语言的接口 .....                   | 141 |
| 4.8.1  | 模块内接口 .....                               | 141 |
| 4.8.2  | 模块间接口 .....                               | 143 |
| 4.9    | 列表文件的使用 .....                             | 146 |
| 4.9.1  | C 语言列表文件 .....                            | 146 |
| 4.9.2  | 汇编语言列表文件 .....                            | 150 |
| 4.10   | $\mu$ Vision2 的使用技巧 .....                 | 153 |
| 4.10.1 | 导入 $\mu$ Vision1 的项目到 $\mu$ Vision2 ..... | 153 |
| 4.10.2 | 为列表文件和目标文件指定单独的文件夹 .....                  | 153 |
| 4.10.3 | 复制工具设置到一个新的目标中 .....                      | 154 |
| 4.10.4 | 使用 $\mu$ Vision2 器件库中没有的微控制器 .....        | 155 |
| 4.11   | $\mu$ Vision2 的高级编程技巧 .....               | 156 |
| 第 5 章  | 用 $\mu$ Vision2 调试项目 .....                | 165 |
| 5.1    | 用 $\mu$ Vision2 调试项目 .....                | 165 |
| 5.1.1  | 设置调试参数 .....                              | 167 |
| 5.1.2  | 指定调试器初始化文件 .....                          | 168 |
| 5.1.3  | 启动代码调试模式 .....                            | 170 |
| 5.1.4  | 使用反汇编窗口 .....                             | 171 |
| 5.1.5  | 使用断点 .....                                | 173 |
| 5.1.6  | 使用变量和函数观察窗口 .....                         | 177 |
| 5.1.7  | 使用 CPU 寄存器观察窗口 .....                      | 181 |
| 5.1.8  | 使用内存观察窗口 .....                            | 181 |
| 5.1.9  | 使用串口观察窗口 .....                            | 183 |
| 5.1.10 | 使用执行效果观察窗口 .....                          | 184 |
| 5.1.11 | 使用内存标记窗口 .....                            | 185 |
| 5.1.12 | 使用符号观察窗口 .....                            | 186 |
| 5.1.13 | 程序的运行 .....                               | 188 |
| 5.2    | 在 Command 窗口中使用调试命令 .....                 | 188 |
| 5.2.1  | 调试命令概述 .....                              | 189 |
| 5.2.2  | 调试命令详解 .....                              | 190 |
| 5.3    | $\mu$ Vision2 调试器表达式 .....                | 212 |
| 5.3.1  | 地址空间及地址空间类型 .....                         | 212 |
| 5.3.2  | 调试常量 .....                                | 213 |
| 5.3.3  | 调试变量 .....                                | 216 |
| 5.3.4  | 调试符号 .....                                | 217 |
| 5.3.5  | 调试表达式 .....                               | 221 |
| 5.4    | $\mu$ Vision2 的调试函数 .....                 | 225 |
| 5.4.1  | 函数分类 .....                                | 226 |



|        |                              |     |
|--------|------------------------------|-----|
| 5.4.2  | 创建和调用函数                      | 231 |
| 5.4.3  | $\mu$ Vision2 调试器的特点         | 236 |
| 5.5    | 一些调试技巧                       | 237 |
| 5.5.1  | 仿真 I/O 端口                    | 237 |
| 5.5.2  | 仿真中断和时钟输入                    | 237 |
| 5.5.3  | 仿真外部 I/O 设备                  | 238 |
| 5.5.4  | 从 PC 串口输入到 8051 串口           | 238 |
| 5.5.5  | 检查非法内存使用                     | 239 |
| 5.5.6  | 从文件读入调试命令                    | 239 |
| 5.5.7  | 预置 I/O 端口和内存的值               | 239 |
| 5.5.8  | 调试结果输出文件                     | 239 |
| 5.5.9  | 使用快捷键                        | 239 |
| 5.5.10 | 内核调试                         | 240 |
| 第 6 章  | $\mu$ Vision2 的实时操作系统 RTX-51 | 241 |
| 6.1    | RTX-51 简介                    | 241 |
| 6.2    | RTX-51 的任务                   | 242 |
| 6.2.1  | RTX-51 单任务程序                 | 242 |
| 6.2.2  | RTX-51 循环任务切换                | 242 |
| 6.2.3  | RTX-51 循环多任务切换               | 242 |
| 6.2.4  | RTX-51 事件和延时                 | 243 |
| 6.2.5  | 使用 RTX-51 信号                 | 244 |
| 6.2.6  | 抢先任务切换                       | 244 |
| 6.2.7  | RTX-51 的其他特性                 | 245 |
| 6.3    | RTX-51 的系统函数                 | 246 |
| 6.3.1  | 函数一览                         | 246 |
| 6.3.2  | 函数详解                         | 247 |
| 6.4    | 使用 RTX-51 Tiny 的要求和限定        | 253 |
| 6.5    | RTX-51 Tiny 的任务管理            | 255 |
| 6.6    | RTX-51 Tiny 的配置文件            | 256 |
| 6.7    | RTX-51 应用实例                  | 258 |
| 6.7.1  | 项目介绍                         | 258 |
| 6.7.2  | 源代码注释                        | 259 |
| 6.7.3  | 交通灯控制器命令                     | 270 |
| 6.7.4  | TRAFFIC.Uv2 项目调试             | 270 |
| 第 7 章  | 在 $\mu$ Vision2 中使用片上外设      | 280 |
| 7.1    | 特殊功能寄存器                      | 280 |
| 7.2    | 通用寄存器组                       | 284 |
| 7.3    | 中断服务程序                       | 285 |
| 7.4    | 并行 I/O 口                     | 290 |

|               |                                                  |            |
|---------------|--------------------------------------------------|------------|
| 7.5           | 定时器/计数器                                          | 292        |
| 7.6           | 串行接口                                             | 294        |
| 7.7           | 看门狗定时器                                           | 295        |
| 7.8           | 数/模转换 (D/A) 和模/数转换 (A/D)                         | 296        |
| 7.9           | 低功耗模式                                            | 298        |
| <b>第 8 章</b>  | <b>使用 <math>\mu</math>Vision2 内嵌的 Monitor-51</b> | <b>300</b> |
| 8.1           | 使用 Monitor-51 的硬软件要求                             | 300        |
| 8.2           | Monitor-51 的使用方法                                 | 302        |
| 8.3           | 配置 Monitor-51                                    | 302        |
| 8.4           | 一个实例                                             | 306        |
| 8.5           | 使用 Monitor-51 的限制                                | 309        |
| 8.6           | 故障诊断                                             | 310        |
| <b>附录</b>     |                                                  | <b>312</b> |
| 附录 A          | C51 的库函数                                         | 312        |
| A.1           | 字符函数 (CTYPE.H)                                   | 313        |
| A.2           | 一般 I/O 函数 STDIO.H                                | 319        |
| A.3           | 字符串函数 STRING.H                                   | 327        |
| A.4           | 标准函数 STDLIB.H                                    | 335        |
| A.5           | 数学函数 MATH.H                                      | 339        |
| A.6           | 绝对地址访问 ABSACC.H                                  | 347        |
| A.7           | 内部函数 INTRINS.H (本征函数)                            | 348        |
| A.8           | 变量参数表 STDARG.H                                   | 350        |
| A.9           | 全程跳转 SETJMP.H                                    | 352        |
| A.10          | 访问 SFR 和 SFR bitd 地址的 REGXXX.H                   | 353        |
| 附录 B          | Keil C51 与 ANSI C 的差别                            | 354        |
| 附录 C          | Keil C51 不同版本的差别                                 | 356        |
| 附录 D          | $\mu$ Vision2 支持的 8051CPU 派生器件                   | 360        |
| 附录 E          | $\mu$ Vision2 的错误信息                              | 362        |
| <b>主要参考文献</b> |                                                  | <b>374</b> |

# 第1章 Keil C51 的简单回顾

## 1.1 51 单片机的特点

C 语言是一种程序语言，针对不同的单片机处理器，相关的 C 语言都会有一些细节的改变。编写 C 语言程序时，如要对硬件编程，就必须对硬件有一定的认识，对 51 单片机编程更是如此，因为它的开发应用是不可与硬件脱节的。首先初步了解一下 51 芯片的结构和引脚功能。MSC51 架构的芯片种类很多，具体特点和功能不尽相同，在此以 Atmel 公司的 AT89C51 和 AT89C2051 为中心对象来进行讲解，两者是 AT89 系列的典型代表，应用资料很多，价格便宜，是应用 51 的首选芯片。

图 1.1 是 AT89C51 和 AT89C2051 的 DIP 形式的引脚封装图（此外，AT89C51 还具

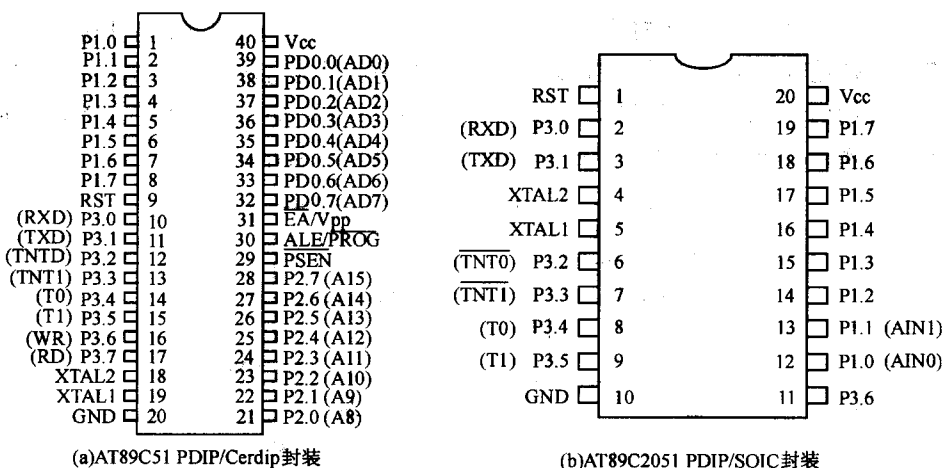


图 1.1 AT89C51 和 AT89C2051 引脚封装图

表 1.1 AT89C51 和 AT89C2051 主要性能表

| AT89C51                       | AT89C2051                     |
|-------------------------------|-------------------------------|
| 4KB 可编程 Flash 存储器（可擦写 1000 次） | 2KB 可编程 Flash 存储器（可擦写 1000 次） |
| 三级程序存储器保密                     | 两级程序存储器保密                     |
| 静态工作频率：0Hz~24MHz              | 静态工作频率：0Hz~24MHz              |
| 128B 内部 RAM                   | 128B 内部 RAM                   |
| 2 个 16 位定时器/计数器               | 2 个 16 位定时器/计数器               |
| 一个串行通信口                       | 一个串行通信口                       |
| 6 个中断源                        | 6 个中断源                        |
| 32 条 I/O 引线                   | 15 条 I/O 引线                   |
| 片内时钟振荡器                       | 1 个片内模拟比较器                    |

有 PLCC、PQFP 和 TQFP 的封装形式), 表 1.1 是它们的主要性能表。可以看出它们是大体相同的, 由于 AT89C2051 的 I/O 线很少, 导致它无法外加数据 RAM 和程序 ROM, 片内 Flash 存储器也少, 但它的体积比 AT89C51 小很多, 虽然它们各有其特点, 但其核心是一样的, 使用者可根据实际需要来选用。下面介绍 AT89C51 和 AT89C2051 的具体功能。

### 1.1.1 AT89C51 的引脚功能

#### 1. 电源引脚

Vcc 40 脚 电源端

GND 20 脚 接地端

工作电压为 5V, AT89LV51 的工作电压为 2.7~6V, 引脚功能一样。

#### 2. 外接晶体引脚

XTAL1 19 脚

XTAL2 18 脚

图 1.2 所示的是 AT89LV51 外接晶体引脚。XTAL1 是片内振荡器的反相放大器输入端, XTAL2 则是输出端, 使用外部振荡器时, 外部振荡信号应直接加到 XTAL1, 而 XTAL2 悬空。在使用内部方式时, 时钟发生器对振荡脉冲二分频, 如晶振频率为 12MHz, 时钟频率就为 6MHz。晶振的频率可以在 1~24MHz 内选择, 电容取 30pF 左右。

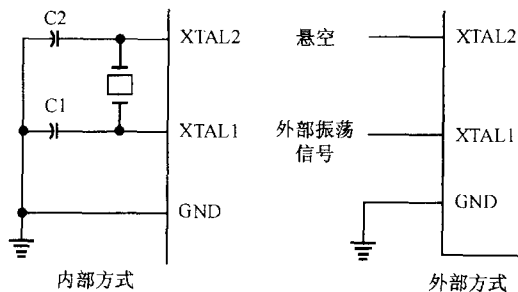


图 1.2 AT89LV51 外接晶体引脚

对于型号同样为 AT89C51 的芯片, 在其后面还有频率编号, 有 12MHz、16MHz、20MHz 和 24MHz 等可选, 在购买和选用时需要注意, 如 AT89C51 24PC 就是最高振荡频率为 24MHz 的普通商用芯片。

#### 3. 复位引脚

RST 9 脚

在振荡器运行时, 有两个机器周期 (24 个振荡周期) 以上的高电平出现在此引脚时, 将使单片机复位, 只要这个脚保持高电平, 51 芯片便循环复位。复位后 P0~P3 口均置 1, 引脚表现为高电平, 程序计数器和特殊功能寄存器 SFR 全部清零。当复位引脚由高电平变为低电平时, 芯片从 ROM 的 00H 地址处开始运行程序, 复位操作不会对内部的数据 RAM 有所影响。常用的复位电路如图 1.3 所示。

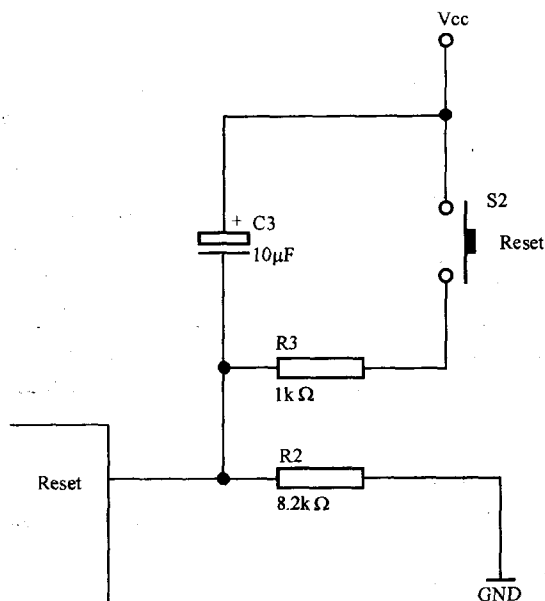


图 1.3 常用复位电路

#### 4. 输入/输出引脚

##### (1) P0 端口 (P0.0~P0.7)

P0 是一个 8 位漏极开路型双向 I/O 端口，端口置 1（对端口写 1）时作高阻抗输入端。作为输出端口时能驱动 8 个 TTL 负载。对内部 Flash 程序存储器编程时，接收指令字节；校验程序时输出指令字节，要求外接上拉电阻。

在访问外部程序和外部数据存储器时，P0 口是分时分转换的地址(低 8 位)/数据总线，访问期间内部的上拉电阻起作用。

##### (2) P1 端口 (P1.0~P1.7)

P1 是一个带有内部上拉电阻的 8 位双向 I/O 端口。输出时可驱动 4 个 TTL 负载。端口置 1 时，内部上拉电阻将端口拉到高电平，作输入用。对内部 Flash 程序存储器编程时，接收低 8 位地址信息。

##### (3) P2 端口 (P2.0~P2.7)

P2 是一个带有内部上拉电阻的 8 位双向 I/O 端口。输出时可驱动 4 个 TTL 负载。端口置 1 时，内部上拉电阻将端口拉到高电平，作输入用。对内部 Flash 程序存储器编程时，接收高 8 位地址和控制信息。

在访问外部程序和 16 位外部数据存储器时，P2 口送出高 8 位地址。而在访问 8 位地址的外部数据存储器时其引脚上的内容在此期间不会改变。

##### (4) P3 端口 (P3.0~P3.7)

P3 是一个带有内部上拉电阻的 8 位双向 I/O 端口。输出时可驱动 4 个 TTL 负载。端口置 1 时，内部上拉电阻将端口拉到高电平，作输入用。对内部 Flash 程序存储器编程时，接收控制信息。除此之外 P3 端口还用于一些专门功能，具体功能如表 1.2 所示。

值得注意的是，P1~P3 端口在作为输入使用时，因内部有上接电阻，被外部拉低的

引脚会输出一定的电流。

表 1.2 P3 端口引脚兼用功能表

| P3 引脚 | 兼用功能          |
|-------|---------------|
| P3.0  | 串行通信输入 (RXD)  |
| P3.1  | 串行通信输出 (TXD)  |
| P3.2  | 外部中断 0 (TNT0) |
| P3.3  | 外部中断 1 (TNT1) |
| P3.4  | 定时器 0 输入 (T0) |
| P3.5  | 定时器 1 输入 (T1) |
| P3.6  | 外部数据存储器写选通 WR |
| P3.7  | 外部数据存储器写选通 RD |

什么叫上拉电阻？简单地说，上拉电阻就是把电平拉高，通常用  $4.7\sim 10.0\text{k}\Omega$  的电阻接到  $V_{cc}$  电源，下拉电阻则是把电平拉低，电阻接到 GND 地线上。

#### 5. 其他的控制或复用引脚

##### (1) ALE/PROG 30 脚

访问外部存储器时，ALE（地址锁存允许）的输出用于锁存地址的低位字节。即使不访问外部存储器，ALE 端仍以不变的频率输出脉冲信号（此频率是振荡器频率的  $1/6$ ，故该引脚通常作为硬件调试时的输出观察）。在访问外部数据存储器时，出现一个 ALE 脉冲。对 Flash 存储器编程时，这个引脚用于输入编程脉冲 PROG。

##### (2) PSEN 29 脚

该引脚是外部程序存储器的选通信号输出端。当 AT89C51 由外部程序存储器取指令或常数时，每个机器周期输出 2 个脉冲即两次有效。但访问外部数据存储器时，将不会有脉冲输出。

##### (3) EA/ $V_{pp}$ 31 脚

外部访问允许端。当该引脚访问外部程序存储器时，应输入低电平。要使 AT89C51 只访问外部程序存储器（地址为  $0000\text{H}\sim\text{FFFFH}$ ），这时该引脚必须保持低电平。对 Flash 存储器编程时，用于施加  $V_{pp}$  编程电压。 $V_{pp}$  电压有两种，类似芯片的最大频率值要根据附加的编号或芯片内的特征字决定。

### 1.1.2 AT89C2051 简介

AT89C2051 是由 Atmel 公司推出的一种小型单片机，1995 年出现在中国市场。其主要特点为采用 Flash 存储器技术，降低了制造成本。其软件、硬件与 MCS-51 完全兼容，很快被中国广大用户接受，其程序可被多次擦写，使得开发与试验比较容易。

#### 1. 引脚

AT89C2051 共有 20 条引脚，从图 1.1 (b) 中可见，AT89C2051 继承了 AT89C51 最重要引脚，由于 1.1.1 节对 AT89C51 的引脚功能进行了详细的介绍，而 AT89C2051 和 AT89C51 的相应引脚在功能上完全一致，这里只简单介绍 AT89C2051 的特有功能。

P1 口共 8 脚, 准双向端口。P3.0~P3.6 共 7 脚, 准双向端口, 并且保留了全部的 P3 的第二功能, 如 P3.0、P3.1 的串行通信功能, P3.2、P3.3 的中断输入功能, P3.4、P3.5 的定时器输入功能。

在引脚的驱动能力上面, AT89C2051 具有很强的下拉能力, P1、P3 口的下拉能力均可达到 20mA。相比之下, AT89C51/AT87C51 的端口下拉能力每脚最大为 15mA。但是限定 9 脚电流之和小于 71mA。这样, 引脚的平均电流只有 9mA。AT89C2051 驱动能力的增强, 使得它可以直接驱动 LED 数码管。

为了增加对模拟量的输入功能, AT89C2051 在内部构造了一个模拟信号比较器, 其输入端连到 P1.0 和 P1.1 口, 比较结果存入 P3.6 对应寄存器 (P3.6 在 AT89C2051 外部无引脚)。对于一些不大复杂的控制电路, 可以增加少量元件来实现, 例如, 对温度的控制, 过压的控制等。

## 2. 存储器

AT89C2051 片内含有 2KB 的 Flash 程序存储器, 128B 的片内 RAM 与 AT89C51 内部完全类似。由于 AT89C2051 内部设计全静态工作, 所以允许工作的时钟为 0~20MHz, 也就是说, 允许在低速工作时, 不破坏 RAM 内容。相比之下, 一般 AT89C51 对最低工作时钟限制为 3.5MHz, 因为其内部的 RAM 是动态刷新的。AT89C2051 不允许构造外部总线来扩充程序/数据存储器, 所以它也不需要 ALE、PSEN、RD 和 WR 一类的引脚。

## 3. 程序保密

AT89C2051 设计有 2 个程序保密位, 保密位 1 被编程之后, 程序存储器不能再被编程, 除非做一次擦除, 保密位 2 被编程之后, 程序不能被读出。

## 4. 软硬件的开发

AT89C2051 可以采用下面 2 种方法开发应用系统。

1) 由于 AT89C2051 内部程序存储器为 Flash, 所以修改它内部的程序十分方便快捷, 只要配备一个可以编程 AT89C2051 的编程器即可。调试人员可以采用程序编辑—编译—固化—插到电路板中试验, 这样反复循环的方法, 对于熟练的 MCS-51 程序员来说, 这种调试方法并不十分困难。

2) 将普通 AT89C51 仿真器的仿真插头中 P1.0~P1.7 和 P3.0~P3.6 引出来仿真 AT89C2051, 这种方法可以运用单步、断点的调试方法, 但是仿真不够真实, 比如 AT89C2051 的内部模拟比较器功能, P1 口、P3 口的增强下拉能力等。

# 1.2 C51 程序的基本结构

C51 源程序结构与一般 C 语言没有什么差别。C51 源程序文件的扩展名为“.c”, 如 Add.c、Max.c 等, 一个 C51 源程序大体上是一个函数定义的集合。在这个集合中有且仅有一个名为 main() 的函数, 也称为该程序的主函数, 主函数是程序的入口, 它是一个特殊的函数, 程序的执行都是从 main() 函数开始的。主函数中的所有语句执行完毕, 则程

序执行结束。一个 C51 源程序必须有且只能有一个名为 main()的函数。

C51 源程序的编程要点如下:

1) C 语言是由函数构成的。一个 C51 源程序至少包含一个 main()函数, 也可以包含一个 main()函数和若干其他函数。因此, 函数是 C51 源程序的基本单位。被调用的函数可以是编译器提供的库函数, 也可以是用户根据需要自己编制设计的函数。

2) 一个 C51 源程序总是从 main()函数开始执行的, 不论 main()函数在整个程序中的位置如何。

3) C51 源程序书写格式自由, 一行内可以写几个语句, 一个语句也可以分写在多行上, C51 源程序无行号。

4) 每个语句和数据定义的最后必须有一个分号, 分号是 C51 源程序语句的必要组成部分。分号不可少, 即使是程序中最后一个语句也应包含分号。

5) C 语言本身没有输入输出语句。输入和输出的操作是由库函数 scanf()和 printf()等函数来完成的。C51 源程序对输入输出实行“函数化”。

6) 可以用 /\*...\*/ 对 C51 源程序中的任何部分作注释。一个好的、有使用价值的程序都应当加上必要的注释, 以增加程序的可读性。

下面是一个简单的 C51 源程序例子 (test.c):

```
# include <reg52.h>          /*C 编译器内部自带的 H 文件, 使用<>*/
void Test1(void);           /*C 函数声明 */

void main(void)
{
    Test1( );               /*调用函数 Test1( )*/
    unsigned char i;        /*C 变量声明*/
    while(1){
        i++;
    }
}

void Test1(void)
{
    unsigned char Work;
    Work++;
}
```

在 C51 源程序中, 常用项目来管理各个文件。项目一般分为两大块: C 文件块和头文件块。常把不同功能写在不同的 C 文件中, 依靠项目的管理, 最后把所有文件连接起来, 这样就可以得到可以烧录的 HEX 文件或 BIN 文件。这些 C 文件中, 有且只有唯一的一个包括 main()函数。可以用头部文件把各个不同的 C 文件互相连接起来。一个 C 文件基本上要对应有一个 H 头部文件, 这个 H 文件就包含本 C 文件中可以提供给外面使用的变量和函数, 没有在 H 文件中列出的文件, 可以算是该 C 文件的内部函数和变量, 外部 C 文件不能使用。



从以上的例子可以看出，C51 源程序一般只有如下的结构：

```
# include < >          /*预处理命令*/
int function_1();       /*函数 1 定义*/
char function_2();      /*函数 2 定义*/
.....
float function_n();     /*函数 n 定义*/
main() {                /*主函数*/
.....
}

function_1(){           /*功能函数 1*/
.....                 /*功能函数 1 函数体*/
}

char function_2(){      /*功能函数 2*/
.....                 /*功能函数 2 函数体*/
}

.....
float function_n(){     /*功能函数 n*/
.....                 /*功能函数 n 函数体*/
}
```

一个 C 语言程序至少应包含一个 main() 函数，也可以有一个 main() 函数和其他的许多功能函数。函数之间可以相互调用，但 main() 函数只能调用其他的功能函数，而不能被其他函数所调用。功能函数可以是 C 语言编译器提供的库函数，也可以由用户按实际需要自行编写。再次说明，不管 main() 函数处于程序中的什么位置，程序总是从 main() 函数开始执行。

### 1.3 C51 的标识符与关键字

标识符是用来标识源程序中某个对象的名字的，这些对象可以是语句、数据类型、函数、变量、数组等。C 语言是大小写字敏感的一种高级语言，如果我们要定义一个定时器 1，可以写为“Timer1”，如果程序中有“TIMER1”，那么这两个是完全不同定义的标识符。标识符由字符串，数字和下划线等组成，注意的是第一个字符必须是字母或下划线，如“1Timer”是错误的，编译时便会有错误提示。有些编译系统专用的标识符是以下划线开头，所以一般不要以下划线开头命名标识符。标识符在命名时应当简单，含义清晰，这样有助于阅读和理解程序。在 C51 编译器中，只支持标识符的前 32 位为有效标识，一般情况下也足够用了，基本上不会出现超过 32 位的标识符。

关键字则是编程语言保留的特殊标识符，它们具有固定名称和含义，在程序编写中不允许标识符与关键字相同。在 Keil  $\mu$ Vision2 中的关键字除了有 ANSI C 标准的 32 个关键字外，还根据 51 单片机的特点扩展了相关的关键字，如表 1.3 和表 1.4 所示。在 Keil  $\mu$ Vision2 的文本编辑器中编写 C51 源程序，系统可以把保留字以不同颜色显示，缺省颜