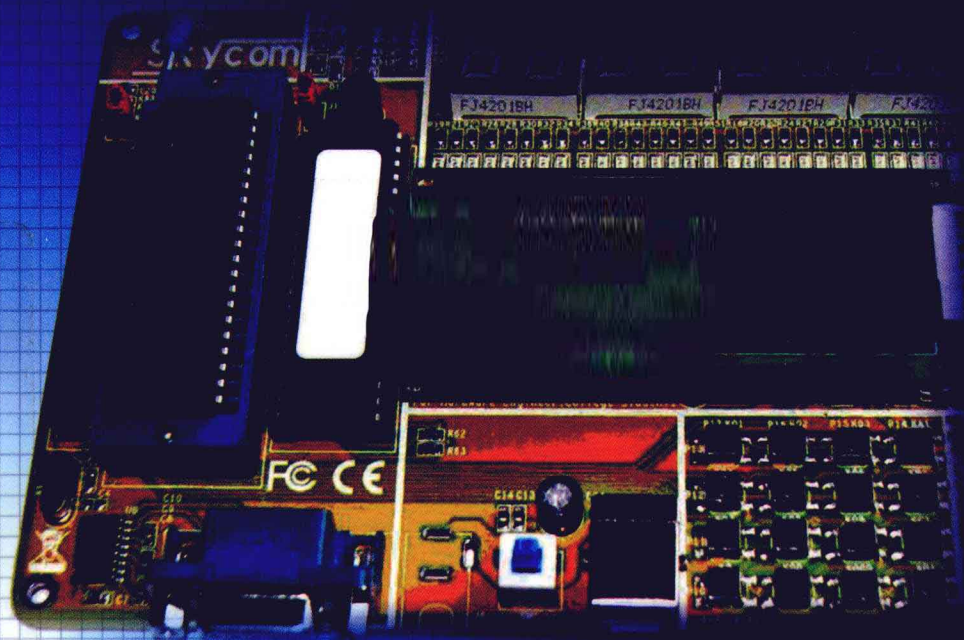


机电专业新技术普及丛书

单片机实用技术

DANPIANJI SHIYONG JISHU

王建 宋永昌 魏福江 主编



机械工业出版社
CHINA MACHINE PRESS



机电专业新技术普及丛书

单片机实用技术

主 编	王 建	宋永昌	魏福江
副主编	王春晖	张 宏	魏晓林
	乔海燕		
参 编	杨玉峰	安亚洲	吴敬萍
	杨亚奇	孙 平	赵颖勋
	郭克亮	王亚须	
主 审	吴 婧		
参 审	汤 瑞	李迎波	



机械工业出版社

本书以实际编程和具体实例为主, 主要内容包括: MCS—51 系列单片机的结构、单片机软件的应用、单片机的基本应用和综合应用。

本书可作为企业培训部门、职业技能鉴定培训机构的培训教材, 还可作为从事单片机应用及开发工作的工程技术人员的参考书。

图书在版编目 (CIP) 数据

单片机实用技术/王建, 宋永昌, 魏福江主编. —北京: 机械工业出版社, 2010

(机电专业新技术普及丛书)

ISBN 978-7-111-32863-6

I. ①单… II. ①王…②宋…③魏… III. ①单片微型计算机—高等学校: 技术学校—教材 IV. ①TP368.1

中国版本图书馆 CIP 数据核字 (2010) 第 254214 号

机械工业出版社 (北京市百万庄大街 22 号 邮政编码 100037)

策划编辑: 朱 华 责任编辑: 赵磊磊

版式设计: 霍永明 责任校对: 张 薇

封面设计: 路恩中 责任印制: 李 妍

北京诚信伟业印刷有限公司印刷

2011 年 3 月第 1 版第 1 次印刷

184mm×260mm·10.75 印张·1 插页·262 千字

0001—3000 册

标准书号: ISBN 978-7-111-32863-6

定价: 25.00 元

凡购本书, 如有缺页、倒页、脱页, 由本社发行部调换

电话服务

网络服务

社服务中心: (010)88361066

门户网: <http://www.cmpbook.com>

销售一部: (010)68326294

销售二部: (010)88379649

教材网: <http://www.cmpedu.com>

读者服务部: (010)68993821

封面无防伪标均为盗版

机电专业新技术普及丛书编委会

主 任：王 建

副主任：楼一光 雷云涛 李 伟 王小涓

委 员：张 宏 王智广 李 明 王 灿 伊洪彬 徐洪亮
施利春 杜艳丽 李华雄 焦立卓 吴长有 李红波
何宏伟 张 桦

前言

FOREWORD

随着经济全球化进程的不断加快,发达国家的制造能力加速向发展中国家转移,我国已成为全球的加工制造基地,但却凸显了我国高技能型人才严重短缺的现实问题,特别是对掌握数控加工技术以及自动化新技术人才的需要越来越多,而很多工人受条件限制,无法到学校接受系统的数控加工技术以及自动化新技术的职业教育;对于离开校园数年、有一定工作经验的人员,也需要进行“充电”,以适应新技术发展的需要。

为解决上述矛盾,本丛书编委会组织一批学术水平高、经验丰富、实践能力强,身处企业、行业一线的专家在充分调研的基础上,结合企业实际需要,共同研究培训目标,编写了这套《机电专业新技术普及丛书》。

本套丛书的编写特色有:

1. 坚持以“以技能为核心,面向青年工人的继续充电、继续提高”为培养方针,把企业和技术工人急需的高新技术进行普及和推广,加快高技能人才的培养,更好地满足企业的用人需求。

2. 更注重实际工作能力和动手技能的培养,内容贴近生产岗位,注重实用,力图实现培训的“短、平、快”,使学员经过培训后能立即胜任本岗位的工作。

3. 在内容上充分体现一个“新”字,即充分反映新知识、新技术、新工艺和新设备,紧跟科技发展的潮流,具有先进性和前瞻性。

4. 以解决实际问题为切入点,尽量采用以图代文、以表代文的编写形式,最大限度降低学习难度,提高读者的学习兴趣。

本套丛书涉及数控技术和电气技术两大领域,是面向有志于学习数控加工、机电一体化以及自动控制实用技术,并从事过相关工作的技术工人的培训用书。适合有一定经验的工人进行自学或转岗培训。

我们希望这套丛书能成为读者的良师益友,能为读者提供有益的帮助!

本书由王建、宋永昌、魏福江任主编,王春晖、张宏、魏晓林、乔海燕任副主编,杨玉峰、安亚洲、吴敬萍、杨亚奇、孙平、赵颖勋、郭克亮、王亚须参加编写。全书由吴婧任主审,汤瑞、李迎波参审。

由于时间和水平有限,书中难免存在不足之处,敬请广大读者批评指正。

编者

目 录

CONTENT

	前言
1	第一章 MCS—51 系列单片机的结构
1	第一节 MCS—51 系列单片机的基本结构
1	一、MCS—51 系列单片机微处理器简介
1	二、8051 单片机的引脚
1	三、8051 单片机各引脚简介
4	第二节 数据及定义
4	一、数据类型
6	二、数值转换
6	三、常量与变量
8	第三节 运算符与语句
8	一、运算符
10	二、语句
16	第四节 函数与指针
16	一、函数
18	二、数组与指针
20	三、C51 程序设计中的技巧
21	第二章 单片机软件的应用
21	第一节 Keil C51 软件的使用方法
21	一、Keil C51 软件的安装方法

22	二、简单程序调试
31	第二节 Easy_51Pro_v20 下载软件的使用方法
31	一、EA51 编程器
32	二、Easy_51Pro_v20 下载软件的安装方法
32	三、Easy_51Pro_v20 下载软件的使用技巧
35	第三节 Proteus ISIS 电子仿真软件的使用方法
35	一、Proteus ISIS 软件的安装方法
41	二、Proteus ISIS 软件的进入方法
41	三、Proteus ISIS 软件的工作界面
42	四、Proteus ISIS 软件的基本操作
49	第三章 单片机的基本应用
49	第一节 I/O 口控制与实验
49	一、MCS—51 系列单片机 I/O 口简介
50	二、I/O 口作为控制输出的实验
54	三、知识扩展
55	第二节 按键扫描与处理
55	一、按键的特性
55	二、程序的处理
56	三、按键扫描实验
60	四、键盘矩阵的处理
61	第三节 蜂鸣器的原理与驱动实验
61	一、蜂鸣器简介
62	二、蜂鸣器实验
64	三、扬声器与功放
66	第四节 数码管动态显示与实验
66	一、数码管简介
67	二、多位数码管的使用及动态扫描显示

68	三、数码管实验
71	四、知识扩展
73	第五节 霍尔开关、光耦合器与中断
73	一、霍尔开关的工作原理
76	二、光耦合器的工作原理
77	三、中断
78	第六节 外部中断输入与技术应用
78	一、设置外部中断输入功能
79	二、计数器实验
80	三、计数中断实验
83	第七节 单片机串行通信与 RS232 接口
83	一、串行通信接口技术
86	二、电平转换芯片 MAX232
87	三、UART 的串行接口结构
88	四、串行通信的软件编程
91	五、串口调试软件的使用
92	六、串口通信实验
97	第八节 段式液晶显示器的工作原理及倒计时牌的制作
97	一、段式液晶显示器简介
97	二、段式液晶显示模块的参数
99	三、段式液晶显示模块的使用
101	四、倒计时牌的制作
106	第九节 字符型液晶显示模块的使用
106	一、单片机液晶显示实验的原理图与电路分析
110	二、字符液晶显示模块库函数的使用
110	三、字符液晶显示实验
114	第十节 模/数转换的原理与实验

115	一、模/数转换的基本原理
117	二、ADC0809 转换芯片
119	三、实验程序
123	第十一节 集成温度传感器 LM35 的使用与实验
123	一、温度传感器简介
124	二、电压型集成温度传感器 LM35 简介
125	三、温度采集实验
131	第四章 单片机综合应用
131	第一节 数字式电压表的制作
131	一、运算放大器的应用
133	二、低频运算放大器 LM358 的原理及应用
134	三、数字式电压表的制作过程
139	第二节 用单片机自动控制水箱给水
139	一、单片机自动给水系统原理图与电路分析
143	二、单片机自动给水工作原理
143	三、软件工作过程及控制程序
145	四、制作要点
145	第三节 LED 汉字显示屏的设计
145	一、LED 点阵工作原理
147	二、LED 汉字显示屏的硬件设计
149	三、字模的提取
151	四、程序设计流程
152	五、实验程序
154	第四节 用单片机改造机床
154	一、用单片机改造机床的原理图与电路分析
156	二、用单片机改造机床的程序编制
161	参考文献

第一章

MCS—51 系列单片机的结构

第一节 MCS—51 系列单片机的基本结构

学习目标

1. 熟悉 MCS—51 系列单片机微处理器的构成及特点。
2. 熟悉 MCS—51 系列单片机的应用范围。

一、MCS—51 系列单片机微处理器简介

一个微型计算机包括微处理器（CPU）、存放程序指令的 ROM 和存放数据的 RAM、输入/输出端口（I/O 口）及时钟、计数器、中断系统等，它们经过地址总线（Address Bus）、数据总线（Data Bus）和控制总线（Control Bus）的连接以及输入/输出端口与外围装置的连接，构成微型计算机系统。由于单片机微处理器把微型计算机的主要部件制造在一块芯片上，所以可以把单片机微处理器看成是一个不带外围装置的微型计算机。

MCS—51 单片机微处理器具有以下特点：

- （1）芯片存储器存储量较小 一般 ROM 小于 4KB 或 8KB，RAM 小于 256B。
- （2）可靠性良好 单片机是按照工业控制要求进行设计的，其抗工业噪声干扰的性能优于一般的 CPU，程序指令及常数数据都烧写在 ROM 内，许多信号通道均在同一个芯片内，因此可靠性高。
- （3）易扩充 单片机具有一般微型计算机所必需的器件，如三态双向总线、并行及串行的输入/输出引脚，可以扩充为各种规模的微型计算机系统。
- （4）控制功能强 为了满足工业控制的要求，单片机指令除了包括输入/输出控制指令、逻辑判断指令外，还有更为丰富的条件分支跳跃指令。

二、8051 单片机的引脚

8051 的引脚如图 1-1 所示。

三、8051 单片机各引脚简介

8051 单片机共有 4 个 I/O 端口，分别为 P0、P1、P2、P3，4 个 I/O 端口都是双向的，并且每个端口都有锁存器。每个端口有 8 条线，因此共有 32 条 I/O 线。各端口的功能叙述如下：

1. P0

P0 有如下三个功能：

- 1) 外部扩充存储器时，当作数据总线 (D0 ~ D7)。
- 2) 外部扩充存储器时，当作地址总线 (A0 ~ A7)。
- 3) 不扩充时，可作一般 I/O 端口使用，但内部无上拉电阻，用于输入或输出时应在外部接上拉电阻。

2. P1

P1 只作 I/O 端口使用，其内部有上拉电阻。

3. P2

P2 有两个功能：

- 1) 外部扩充存储器时，当作地址总线 (A8 ~ A15)。
- 2) 作一般 I/O 端口使用，其内部有上拉电阻。

4. P3

P3 有两个功能，除了作为 I/O 端口使用外（内部有上拉电阻），还有一些特殊的功能，端口引脚的特殊功能见表 1-1，其功能由特殊寄存器来设置。

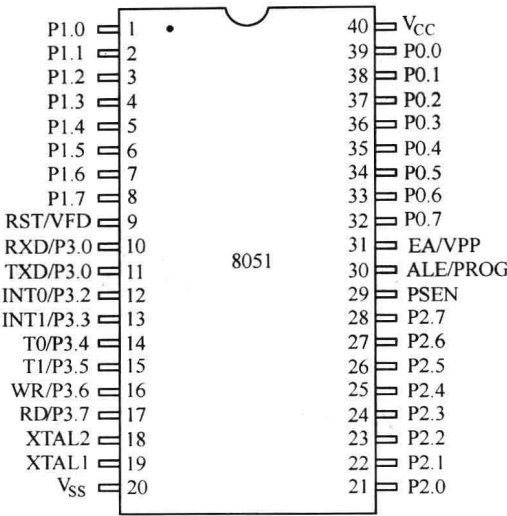


图 1-1 8051 单片机的引脚

表 1-1 端口引脚的特殊功能

端 口 引 脚	特 殊 功 能
P1.0 (8052)	T2 TIMER2 的外部输入引脚 (8052)
P1.1 (8052)	T2EX (TIMER2) 的捕捉 (Capture) / 重新加载 (Reload 的触发输出)
P3.0	RXD (串行输入/输出)
P3.1	TXD (串行输入/输出)
P3.2	INT0 (外部中断)
P3.3	INT1 (外部中断)
P3.4	T0 (TIMER0 的外部输入脚)
P3.5	T1 (TIMER1 的外部输入脚)
P3.6	WR (外部数据存储器的写入控制信号)
P3.7	RD (外部数据存储器的读取控制信号)

端口 P1、P2、P3 有内部上拉电阻，当用于输入时，其电位被拉高，若输入为低电平则可提供电流源；当用于输出时，可驱动 4 个 LS 型 TLL。而端口 P0 用于输入时，处在高阻抗的状态。

其输出缓冲器可驱动 8 个 LS 型 TLL（需要外部的上拉电路）。

5. V_{cc}和 V_{ss}

V_{cc}：电源，+5V。

V_{SS} : GND, 接地。

6. RESET

此脚为高电平时（约两个机器周期），可将 CPU 复位，CPU 复位后其累加器和寄存器的内容见表 1-2。

表 1-2 CPU 复位后其累加器和寄存器的内容

寄 存 器	二 进 制 值
ACC	00000000
B	00000000
PSW	00000000
SP	00000111
P0/P1/P2/P3	11111111
IP	xxx00000
IE	0xx00000
TMOD	00000000
TCON	00000000

7. ALE/PROG

此脚为地址锁存使能信号端，有以下三种功能：

1) 当 8051 外接 RAM/ROM 时，ALE 接地址锁存器 8252 (8212) 的 STB 脚和 74373 的 EN 脚；当 CPU 对外部存储器进行存取时，用以锁住地址的低位地址。

2) 当 8051 未外接 RAM/ROM，在系统中未使用外部存储器时，ALE 脚也会有石英晶体振荡频率的 1/6 输出，可作为外部时钟。

3) 在烧写 EPROM 时 ALE 作为烧写时钟的输入端。

8. PSEN

此脚为程序储存使能端，有以下四种功能：

1) 内部程序存储器读取时不动作。

2) 外部程序存储器（ROM）读取时在每个机器周期内活动两次。

3) 外部数据存储器（RAM）读取时两个 PSEN 脉冲被跳过而不会输出。

4) 外接 ROM 时与 ROM 的 OE 脚连接。

9. EA/VPP

1) 接高电平时：

① CPU 读取内部程序存储器（ROM），如 8051/8052。

② 用于扩充外部 ROM。当读取内部程序存储器超过 0FFFH (8051)、1FFFH (8052) 时，自动读取外部 ROM。

2) 接低电平时 CPU 读取外部程序存储器（ROM），如 8031/8032。

3) 8051 烧写内部 EPROM 时，利用此脚输入 21V 的烧写电压。

10. XTAL1 和 XTAL2

这两个引脚接石英晶体振荡器。

机器周期 = 晶振频率/12。

第二节 数据及定义

学习目标

1. 熟悉 C51 语言的特点。
2. 掌握 C51 语言的语句格式。
3. 掌握并且记忆常用的 C51 语言指令。

程序设计语言分为机器语言、汇编语言和高级语言三大类。

机器语言的每条指令都是由 0、1 组成的序列，每一条语言代表机器可以执行的一个基本操作。机器语言是计算机唯一能够直接识别和执行的，它的执行速度快、效率高。但是机器语言编出的程序难读、难记、难修改，也不具有通用性。

汇编语言是符号化的机器语言。一般情况下，它的指令和机器语言的指令是一一对应的。因为汇编语言仍然是针对硬件、寄存器、内存等进行操作，所以还是把它归为低级语言。同机器语言相比，它的指令可读性更高。但是计算机并不能直接识别和执行汇编语言的指令，必须用汇编编译程序将每一条指令翻译成机器语言指令，计算机才能执行。

机器语言和汇编语言都是面向机器的语言，它们并不符合人们的自然语言，这样就出现了高级语言。高级语言可以用更接近于人类思维的方式来描述算法程序，而不必关心具体的硬件内容。如 Pascal 语言、Cobol 语言、C 语言都属于高级语言。高级语言也需要经过编译转化为机器语言才能为计算机所接受和执行。

C51 编译的语言与 C 语言在很大程度上是一致的，只是针对单片机的自身特点做了一些特殊的扩充。

一、数据类型

任何程序设计都离不开对数据的处理，一个程序如果没有数据，就没有办法工作。

数据在计算机内存中的存放情况是由数据类型决定的。在 C 语言中分为基本数据类型和复杂数据类型，复杂数据类型是由基本数据类型转化而来的。因为复杂数据类型在单片机程序中较少使用，所以只介绍基本数据类型。

C 语言基本数据类型有：char、int、short、long、float 和 double。对于 C51 来说编译时 short 和 int 类型相同，long 和 double 类型相同，所以可以统一标识，这样只剩下 char、int、long 和 float，其中前 3 种还有有符号和无符号之分，表示数值的范围不同，默认情况下都是有符号数，无符号数的表示则要在数据类型前加 unsigned。

1. char（字符类型）

它表示的数据的长度是一个字节，也就是 8 位数。有符号数（char）表示数值范围是 $-128 \sim +127$ 。无符号数（unsigned char）表示数值范围是 $0 \sim 255$ 。

2. int（整型）

它表示的数据的长度是两个字节，也就是 16 位数。有符号数（int）表示数值范围是

-32768 ~ +32767。无符号数 (unsigned int) 表示数值范围是 0 ~ 65535。

3. long (长型)

它表示的数据的长度是四个字节，在单片机中这种类型较少使用，尤其是在 8 位单片机中。

4. float (浮点型)

它是单精度浮点数据类型。一个 float 类型的数据要占用 4 个字节的内存。它没有有无符号之分。浮点型见表 1-3。

表 1-3 浮点型

字节地址	+0	+1	+2	+3
浮点数内容	S EEEEEEE	E MMMMMMM	MMMMMMMM	MMMMMMMM

表 1-3 中，S 为符号位，0 表示正，1 表示负；E 部分为阶码，有 8 位，正常取值范围是 1 ~ 254；M 部分为尾数的小数部分，尾数的整数部分永远为 1，是隐含表示的。一个浮点数值值的计算方法如下：

$$(-1)^S \times 2^{E-127} \times (1.M)$$

它可以表示的数值范围很大，一般都是用在计算数值转换上，很少使用它去定义一个变量，尤其是全局变量，因为它占用内存太多，而且计算消耗的时间太长。比如说要表示一个小数 0.25，如果不是单片机，使用浮点数就可以定义了，而单片机的原则是用最少的资源，所以可以用整型数来表示，也就是把它放大 100 倍，在参与运算时，再把结果缩小至 1/100 就行了，这样就不涉及浮点运算了。除了以上四种基本数据类型以外，还有以下一些数据类型：

5. * (指针型)

指针型数据不同于以上四种基本数据类型。它本身是一个变量，但在这个变量中存放的不是普通的数据，而是指向另一个数据的地址。它也要占用一定的内存单元。在 C51 中，指针的长度一般为 1 ~ 3 个字节。

指针变量也是具有类型的，它的类型表示该指针所指向的地址中数据的类型，比如：char * pt1 表示 pt1 是一个字符型指针变量，也就是 pt1 所指向的地址中的数据类型是字符型。

6. bit(位类型)

这是 C51 编译器的一种扩展类型。在 51 单片机的内存区域有一块 128 位的内存单元，其中每个位都可以定义成位变量，但不能定义为位指针或位数组。

7. sfr(8 位特殊功能寄存器类型)

这也是 C51 编译器的一种扩展类型，利用它可以定义单片机中所有 8 位特殊功能寄存器。

8. sfr16 (16 位特殊功能寄存器类型)

利用它可以定义单片机中所有 16 位特殊功能寄存器。

9. sbit(可寻址位)

单片机中的特殊功能寄存器有许多是可以位寻址的，这些可位寻址的寄存器都有一个特点，就是它们的地址可以被 8 整除。比如 P0，它的地址是 0x80，可以这样定义：

```
sfr    P0 = 0x80;           //将 P0 端口地址定义为 80H
```

```
sbit    P02 = P0^2;           //将 P0.2 位定义为 P02
```

二、数值转换

C 语言允许任何标准数据之间进行数值转换，可以强制转换也可以隐式转换。

1. 强制转换

在要转换的数据前加括号，如 (unsigned char) tmp1 就是将 tmp1 的数据类型强制转换为无符号字符型。如果 tmp1 是整型数 0x1234，那么强制转换后的数将失去高 8 位，成为 0x34。再如 (unsigned int) tmp2，如果 tmp2 是字符型数 0x12，那么强制转换后的数将用 0 补充高 8 位，也就是成为 0x0012。

2. 隐式转换

如果在两种不同类型数据间赋值，C 语言将按照一种优先级别自动将数据进行转换，如：

```
bit→char→int→long→float      signed→unsigned
```

比如定义了下面两种变量：

```
int a; char b;
```

如果进行如下赋值运算：a = b；

则 b 首先由隐式转为整型，再将转换后的数值赋给 a，如果 b = 0x12，则 a = 0x0012。

反过来如果 b = a；

则 a 首先由隐式转为字符型，再将转换后的数值赋给 a，如果 a = 0x1234，则 b = 0x34。

上面箭头表示转换时由低向高进行，并不表示数据必须按这个顺序进行转换，例如一个 bit 类型数据要转换为 int 型，不是先转换为 char 型再转换为 int 型，而是直接转换为 int 型。

在进行数值转换的时候要注意，将大类型数值转为小类型数值时是要丢掉一些数的。

三、常量与变量

1. 常量

常量的值在程序运行中是不能被改变的，常量也是有数据类型的。无论是常量还是变量，在使用前都是要先进行定义。常量的定义格式是：

```
#define    常量名 常量数值
```

例如：

```
#define CONST1    30
```

注意：语句的后面是不加分号的。因为常量数值在整个程序中都是不改变的，所以一般把常量的定义放在程序的开头。定义后的常量就可以在程序中被使用。

2. 变量

变量是一种在程序执行过程中，其值能够不断变化的量。变量的定义格式是：

数据类型 [存储器类型] 变量名表；

要根据变量的取值范围来定义合适的数据类型，存储类型可确定变量存放在单片机中的存储空间。

单片机在物理上有 4 个存储空间，分别是片内数据存储器、片外数据存储器、片内程序存储器和片外程序存储器，并且有各自的寻址方式。可以指明定义的数据是属于哪个存储空间，如果不指名，C51 编译器将按一种默认约定对其进行定位。

C51 的存储器类型有 6 种，其中 4 种是常用的：

(1) data 寻址空间为片内数据存储器低位的 128 字节，寻址方式为直接寻址，访问速度最快。

(2) idata 寻址空间为整个片内数据存储器，寻址方式为间接寻址，速度较慢。

(3) xdata 寻址空间为整个片外数据存储器，对应汇编指令 MOVX@ DPTR，属于通过数据指针的访问。

(4) code 指的是程序存储器空间，用 MOVC@ A + DPTR 指令访问。

如果了解单片机存储结构和指令系统，则上面的内容很好理解。如果不了解，可按下面的原则来定义：

1) 如果定义的是需要频繁改变数值的变量，就把它的存储器类型用 data 来规定。

2) 如果 128 字节的数据存储器空间充足，也用 data 存储器类型。

3) 如果 128 字节的数据存储器空间不够，可以把一些不频繁改变的变量或一些占用空间比较大的变量规定为 idata 存储器类型。

4) 多数情况下不需要扩展片外数据存储器，所以 xdata 不经常使用。只有当一些大的图形数据需要显示，而片内数据存储器不够使用，需要扩展片外数据存储器时，才可以把图形数据指定在 xdata 中。

5) 对于有些数据，如汉字字型、英文和字符字型、线性化表格等，这些数据的数据量比较大，但是数据值不变，不可能用常量——表示，此时，可以把它们存放在程序存储器中，编译器会把它们和程序代码编译在一起，通过编程器一同写在单片机的程序存储器中。

存储器类型是用方括号括起的，表示可以不规定，由 C51 编译器自动根据在编译时使用的存储器模式来进行分配，一般使用的是 SMALL 模式。变量都是定义在片内数据存储器中的，先按 data 型存储，data 不足时按 idata 型存储。

例如：

```
unsigned char adval1, adval2;    /* AD 转换值变量 */
int realval;                    /* 实际电压值 */
unsigned char idata lcddata[5]; /* 液晶显示缓冲区 */
/* 字符表 0~9 */
unsigned char code chartab[10] = {
0x88, 0xbb, 0xc1, 0x91, 0xb2, 0x94, 0x84, 0xb9, 0x80, 0x90};
```

第一行定义了两个无符号字符型变量 adval1 和 adval2，它们将被存储在片内数据存储区，用直接寻址方式寻址。第二行定义了一个整型变量 realval，它也将被分配到片内数据存储区，用直接寻址方式寻址。第三行定义了一个 5 元素的数组变量，每个数组元素都是无符号字符型数据，5 个数据的存储地址是连续的，它们将被存储在片内数据存储区，使用间接寻址方式寻址。第四行是在数码管上显示的 0~9 的数字字型，它的值都是不改变的，把它们分配到程序存储器中。

常量定义每行只能定义一个常量，并且后面没有分号，一般常量名大写。变量定义时同类型的变量定义可以放在一起，也可以不放在一起，定义结尾要有分号。一般变量名使用小写。C 语言是区分大小写的，比如 tmp1 和 Tmp1 就是两个不同的变量。

常用到的还有变量的定义：

1) 位变量定义: `bit flag_busy = 0;`

位变量只有两个取值, 即 0 和 1, 一般定义位变量是为了做标志。

2) 可寻址位变量定义: `sbit SCL = P2^0;`

需要把可位寻址的特殊寄存器的某些位单独定义出来, 比如 I/O 口的某个引脚。P2 可进行位寻址, 它有 8 个引脚, 每个引脚都可以单独定义出来, 定义的方法就是按上面的格式。

3) 指针型变量定义: `unsigned char * pt1;`

这里定义了一个指向片内数据存储区数据的指针 `pt1`, 它指向的地址的数据是无符号字符型, 不要理解成指针的类型是无符号字符型。注意变量名前面是有一个 “*” 的。

下面看一段程序:

```
#define CONST1 20
#define CONST2 3.14
int tmp1, tmp2;
float idata tmp3;
void main(void)
{
    tmp1 = 300;
    tmp2 = tmp1 * CONST1;
    tmp1 = 10;
    tmp3 = tmp1 * CONST2;
}
```

运行后的结果如下:

`tmp1 = 10; tmp2 = 6000; tmp3 = 31.4。`

第三节 运算符与语句

学习目标

1. 熟悉 C51 程序设计基本知识。
2. 掌握 C51 运算方法。
3. 掌握 C51 语句格式。

一、运算符

C 语言有丰富的运算符, 正是这些运算符使得对数据的操作非常灵活, 编译成目标代码后, 具有较高的代码效率。

1. 赋值运算符 (=)

赋值运算符用来把一个常量或变量或运算式的值传递给另一个变量。如: `tmp1 = 30;` 就是把值 30 赋值给变量 `tmp1`。