- 以任务驱动为主线，通过14项任务，全面系统地讲解单片机的知识与应用
- 注重实践能力的培养，通过完成单片机的14项任务，强化动手能力
- 突出技能训练，使读者在完成任务项目的实践中掌握单片机控制与应用的技能

单片机控制与应用实训教程

主　编　杨旭方
副主编　李　慧　余金栋

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

单片机控制与应用实训教程

主　编　杨旭方

副主编　李　慧　余金栋

電子工業出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

本书以任务驱动为主线，结合考证需要精心设计任务项目。以必需、够用为原则，注重工程实践，强化动手能力的培训。附有多套考证样题，适合不同层次读者的需求。

全书共设置了 14 项任务，通过对 14 项制作任务项目的讲解，让读者掌握单片机的基本知识、输入/输出端口应用、键盘接口技术、中断原理及应用、定时/计数器原理及应用、数码管静态显示、数码管动态显示、LED 点阵显示、A/D 转换、D/A 转换、串行端口通信原理及应用、I^2C 总线技术、单片机应用系统设计以及步进电机控制等相关知识，重点突出了各项技能的实训。

本书以培养读者对单片机的应用能力为宗旨，突出基础知识的掌握和实践技能的训练，充分体现了职业院校为国家培养技能型人才的特点。

本书可作为职业技术院校及专业培训的教材，也适合从事单片机开发的技术人员阅读。

图书在版编目（CIP）数据

单片机控制与应用实训教程 / 杨旭方主编．—北京：电子工业出版社，2010.5
ISBN 978-7-121-10784-9

Ⅰ．①单… Ⅱ．①杨… Ⅲ．①单片微型计算机－计算机控制－教材 Ⅳ．①TP368.1

中国版本图书馆 CIP 数据核字（2010）第 077561 号

策划编辑： 谭佩香
责任编辑： 鄂卫华
印　　刷： 北京市天竺颖华印刷厂
装　　订： 三河市鑫金马印装有限公司
出版发行： 电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编 100036
开　　本： 787×1092　1/16　印张：14　字数：341 千字
印　　次： 2010 年 5 月第 1 次印刷
定　　价： 28.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。

编委会名单

主　任：崔险峰

副主任：陈　键

成　员：杨旭方　　李　慧　　余金栋　　陈朝大

赖友源　　徐　胜　　杨宝源　　黄荣祥

谭丽明　　黄晓云　　傅秀丽　　黄庆辉

李淑贞　　张　莉　　沈仕宗　　邹彩梅

前言

PREFACE

本书的内容融合了作者多年教学实践和科研工作积累的经验，是作者多年课程教学成果的体现，本书的编写特色有以下几点。

一、全书以任务为主线，通过完成任务来带动教学

本教材的编写以布置任务、分析任务、探索知识和完成任务为主线，将知识点融入到14项单片机的任务项目中，让读者在完成任务项目的技能训练中掌握知识，并培养分析问题和解决问题的能力。

二、结合考证需要，精心设计任务的内容

本书结合单片机快速开发专项能力认证和实际教学要求，精心设计任务项目，力求内容符合考试大纲要求。同时为了降低学习难度，将学习重点与难点巧妙地包含在各项任务之中，力求让读者在完成每一项任务项目的实践中解决2～3个技术难点。

三、以必需够用为原则，注重工程实践

全书以任务驱动为主线，以实际需要为目的来组织、安排章节内容，以必需够用为原则，摒弃过时、应用不多且难度较大的内容，力求内容能满足上岗、教学需要，真正做到学习与就业的无缝对接。

四、强化动手能力培养，适合不同层次读者需求。

全书所有任务制作步骤简洁明了，读者可根据书中操作提示完成任务，通过完成任务，培养读者实际操作能力。

本书主编为杨旭方，副主编为李慧、余金栋，参加编写的还有陈朝大、赖友源、徐胜、杨宝源、崔险峰、陈键、黄荣祥、谭丽明、黄晓云、傅秀丽、黄庆辉、李淑贞、张莉、沈仕宗、邹彩梅等。其中杨旭方编写了任务一、任务六、任务七、任务九和任务十二，李慧编写了任务十、任务十三和附录，陈朝大编写了任务二和任务十一，赖友源编写了任务五和任务八，徐胜编写了任务十四，杨宝源参与部分内容整理。全书由杨旭方统稿，并得到了崔险峰、陈键等老师的大力帮助，在此表示感谢！

由于作者水平有限，书中难免有不妥之处，恳请专家和读者批评指正。

图书联系方式：tan_peixiang@phei.com.cn.

编　者

2010年4月

目　　录

项目一　单个彩灯闪烁

——单片机入门知识

教学目的

掌握：单个彩灯接口电路设计方法；LED 发光二极管的基本使用方法。
理解：单个彩灯闪烁的程序编写方法。
了解：单片机的内部结构及相关知识。

愿你知多点：

在日常生活中，我们经常使用各种 LED 显示器，例如路灯、酒店霓虹灯、市场广告灯等，那么这些 LED 显示器是如何制作的呢？用什么元器件控制？在这一章中，我们将通过完成单个彩灯闪烁这一任务来学习制作 LED 显示器的方法及单片机的相关知识。图 1-1 所示为 LED 发光二极管的应用实例。

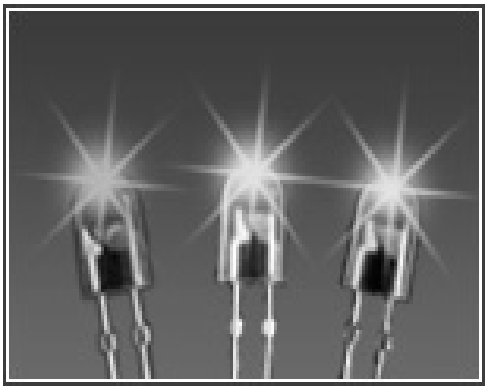


图 1-1　LED 发光二极管的应用实例

1.1　能力培养

通过完成单个彩灯闪烁这一任务，可以培养以下能力。

1. 能识别 LED 发光二极管的引脚。
2. 能正确使用 LED 发光二极管。
3. 能制作单个彩灯。

1.2 任务分析

要完成此项任务，需要掌握以下 5 方面知识。

1. 如何使用 LED 发光二极管。
2. 如何使用二进制数和十六进制数。
3. 如何使用单片机。
4. 如何设计 LED 发光二极管与单片机接口电路。
5. 如何设计单个彩灯闪烁程序。

1.3 如何使用 LED 发光二极管

图 1-2 所示为 LED 发光二极管实物，图 1-3 所示为其电路符号。从图 1-2 和图 1-3 中可以看出，LED 发光二极管与普通二极管一样，共有两只引脚，其中 1 为正极（也称阳极），2 为负极（也称阴极），内部也由一个 PN 结组成，且具有单向导电性（即正向导通，反向截止），但其导通开启电压值比普通二极管高，一般为 1.2～2.5 V。

图 1-2 LED 发光二极管实物

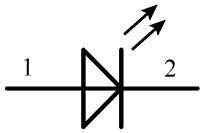


图 1-3 LED 发光二极管电路符号

当 LED 发光二极管正向导通时，二极管点亮，发光亮度的大小与材料、结构以及工作电流有关。一般来说，工作电流越大，二极管的发光亮度越大；当 LED 发光二极管反向截止时，二极管灭，因此只要控制好 LED 发光二极管两端的电压大小，就可以实现单个彩灯的闪烁控制。

1.4 如何使用二进制数和十六进制数

1.4.1 数制

所谓数制，就是人们利用符号计数的一种科学方法，数制有很多种，计算机常用的数制有二进制、十六进制和十进制 3 种。一个数值，可以用不同数制表示。在日常生活中，人们经常使用十进制数，而在对计算机的应用中，主要使用二进制数，但因为二进制数由一长串 0 和 1 组成，位数太多，不便于书写和记忆，所以我们在编程时主要使用十六进制数。

1. 十进制数（Decimal）

十进制数共有 10 个数字符号，它们分别为 0、1、2、3、4、5、6、7、8、9，这 10 个数字符号又称数码。十进制数的主要特点如下所示。

（1） 有 0～9 共 10 个数码。

（2） 基数为 10，逢十进一。

（3） 十进制数用 D 表示。

任何一个十进制数都可以按权展开方式表示，表示方法如下所示。

$$D=A_{n-1}\times10^{n-1}+\cdots+A_2\times10^2+A_1\times10^1+A_0\times10^0+A_{-1}\times10^{-1}+A_{-2}\times10^{-2}+\cdots$$

其中 A_n 表示十进制数的第 n 位；10^n 表示十进制数的第 i 位的权。例如十进制数 35 可表示为

$$35D=3\times10^1+5\times10^0$$

2. 二进制数（Binary）

二进制数只有两个数码，即 0 和 1，特点是逢二进一。二进制数用 B 表示。

同理，任何一个二进制数都可以按权展开方式表示，表示方法如下所示。

$$D=A_{n-1}\times2^{n-1}+\cdots+A_2\times2^2+A_1\times2^1+A_0\times2^0+A_{-1}\times2^{-1}+A_{-2}\times2^{-2}+\cdots$$

其中 A_n 为二进制数的第 n 位，2^n 为二进制数的第 i 位的权。例如二进制数 1011 可表示为

$$1011B=1\times2^3+0\times2^2+1\times2^1+1\times2^0$$

3. 十六进制数（Hexadecimal）

十六进制数共有 16 个数码，即 0、1、2、3、4、5、6、7、8、9、A、B、C、D、E 和 F，其特点是逢十六进一。十六进制数用 H 表示。

十六进制数也可以按权展开方式表示，方法与十进制数类似，表示方法如下所示。

$$D=A_{n-1}\times16^{n-1}+\cdots+A_2\times16^2+A_1\times16^1+A_0\times16^0+A_{-1}\times16^{-1}+A_{-2}\times16^{-2}+\cdots$$

其中 A_n 为十六进制数的第 n 位，16^n 为十六进制数的第 i 位的权。例如十六进制数 17 可表示为

$$17H=1\times16^1+5\times16^0$$

温馨提示：

以字母开头的十六进制数在书写时要在字母前加 0，如 BH 要写成 0BH。

1.4.2 数制转换

1. 二进制数与十进制数之间的转换

（1） 二进制数转换成十进制数

要将二进制数转换成十进制数，只要将二进制数按权展开后相加即可。例如

$$1011B=1\times2^3+0\times2^2+1\times2^1+1\times2^0=11D$$

（2） 十进制数转换成二进制数

将十进制数转换成二进制数通常采用“除 2 取余倒记法”，具体方法是先用十进制数连续除以 2，直到所得商小于 2 为止，然后再将各次所得的余数按逆序写成二进制数形式。

例如，将 15H 转换为二进制数，转换过程为

```
2 | 15 | 1
  2 | 7 | 1
    2 | 3 | 1
      | 1 |
```

温馨提示：

“除 2 取余倒记法”适用于十进制数的整数形式转换成二进制数的整数形式，若十进制数为小数，则采用“乘 2 取整顺记法”。

例如，将 0.75H 转换为二进制数，转换过程为 0.75×2=1.5，取 1.5 的整数部分为 1；1.5 去除整数部分后为 0.5，而 0.5×2=1.0，则 0.75D=0.11B。

2. 十六进制数与十进制数之间的转换

（1） 十六进制数转换成十进制数

要将十六进制数转换成十进制数，只要将十六进制数按权展开后相加即可。例如

$$15H=1\times16^1+5\times16^0=21D$$

（2） 十进制数转换成十六进制数

将十进制数转换成十六进制数通常采用“除 16 取余倒记法”，具体方法是先用十进制

数连续除以 16，直到所得商小于 16 为止，然后再将各次所得的余数按逆序写成十六进制数形式。

例如，将 35D 转换为十六进制数，转换过程为

```
16 | 35    3
     2     2
```

温馨提示：

“除 16 取余倒记法”适用于十进制数的整数形式转换成十六进制数的整数形式，若十进制数为小数，则采用“乘 16 取整顺记法”。

例如，将 0.625 转换为十六进制数，转换过程为 0.625×16=0AH，取 0A 的整数部分为 A，则 0.75D=0.AH。

3. 十六进制数与二进制数之间的转换

（1） 二进制数转换成十六进制数

将二进制数转换成十六进制数通常采用“四合一法”，方法如下所示。

① 整数部分从二进制数的低位（即小数点左侧）开始，每 4 位作为一组划分整数部分，不足 4 位左补 0。

② 小数部分从二进制数的高位（即小数点的右侧）开始，每 4 位作为一组划分小数部分，不足 4 位右补 0。

例如，将 101.11B 转换成十六进制数。

```
0101.1100
   5    C
```

则，101.11B=5CH。

（2） 十六进制数转换成二进制数

将十六进制数转换成二进制数通常采用“一分为四法”，方法如下所示。

十六进制数的整数部分和小数部分的每 1 位均用 4 位二进制数表示，然后再删除整数部分最左侧和小数部分最右侧多余的 0。

例如，将 3A.5H 转换成二进制数。

```
   3     A . 5
0011  1010 . 0101
```

1.5　如何使用单片机

1.5.1　单片机简介

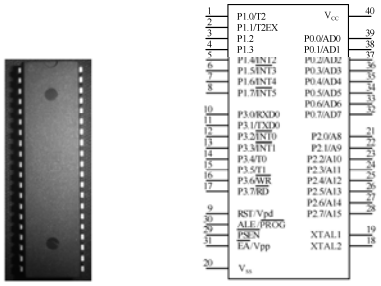
单片微型计算机（Single-Chip Microcomputer）简称单片机，是微型计算机发展的一个

重要分支。它采用超大规模技术将具有数据处理能力的微处理器（CPU）、随机存储器（RAM）、只读存储器（ROM）、输入/输出接口电路等集成到一个芯片上，构成一个微型计算机系统。因单片机具有体积小、功能强、可靠性高、功耗低等优点，在家用电器、自动化仪表、工业控制等领域得到了广泛的应用。

单片机的应用改变了传统控制系统的设计理念。传统控制系统中的继电器、可控硅以及其他电路的控制，在单片机系统中将由软件实现。软件控制既节省成本，又提高控制系统的可靠性。

1.5.2 MCS—51 系列单片机的引脚

MCS—51 系列单片机常见的封装方式有双列直插式封装（DIP）和方形扁平式（QFP）封装两种，下面以 MCS—51 系列单片机中用双列直插式封装的 40 脚芯片为例，来介绍单片机的引脚。图 1-4 所示为单片机的实物图和引脚图。



（a）实物图　　（b）引脚图

图 1-4　MCS—51 系列单片机的实物图和引脚图

1. 电源引脚 Vcc 和 Vss

Vcc：电源电压输入端，一般为+5 V。

Vss：接地端。

2. 时钟引脚 XTAL1 和 XTAL2

在使用内部振荡电路时，XTAL1 和 XTAL2 用于外接晶体振荡器和微调电容器，晶体振荡器频率即为振荡频率；在使用外部时钟时，XTAL1 和 XTAL2 用于外接时钟源。

3. 控制信号引脚 RST/Vpd、ALE/$\overline{\text{PROG}}$，$\overline{\text{PSEN}}$和$\overline{\text{EA}}$/Vpp

（1） RST/Vpd

RST 为复位信号输入端。当 RST 端保持两个机器周期（24 个时钟周期）以上的高电平时，单片机完成复位操作。Vpd 为单片机内部 RAM 的备用电源电压输入端。当主电源电压发生断电（也称为掉电或失电）或电压降到一定值时，可以通过 Vpd 端为单片机内部 RAM 提供备用电源电压，以保护单片机内部 RAM 中的信息不丢失，并且在接上主电源电压后能够继续正常运行。

（2） ALE/$\overline{\text{PROG}}$

ALE 为地址锁存信号端。当访问外部存储器时，ALE 端用于锁存 P0 端口输出的低 8 位

地址信号。在不访问外部存储器时，ALE 端可以输出为 1/6 晶体振荡器频率的脉冲信号，该脉冲信号可以用于外部定时。$\overline{\text{PROG}}$ 为单片机内部 EPROM 编程时的编程脉冲信号输入端。

（3）$\overline{\text{PSEN}}$

$\overline{\text{PSEN}}$ 为外部程序存储器的读选通信号输入端，当访问外部 ROM 时，$\overline{\text{PSEN}}$ 产生负脉冲信号作为外部 ROM 的选通信号。

（4）$\overline{\text{EA}}$ /Vpp

$\overline{\text{EA}}$ 为访问外部程序存储器的控制信号输入端。当 $\overline{\text{EA}}$ 端输入为低电平时，CPU 访问外部程序存储器，当 $\overline{\text{EA}}$ 端输入为高电平时，CPU 访问内部程序存储器。由于图 1-4 中所示的单片机没有内部 ROM，因此 $\overline{\text{EA}}$ 必须接地。Vpp 为单片机内部 EPROM 的 21 V 编程电源电压输入端。

4. 4 个 8 位的 I/O 端口 P0、P1、P2、P3

（1） P0 端口（P0.0～P0.7）

P0 端口的第一功能是作为一个 8 位漏极开路型的双向 I/O 端口。第二功能是在访问外部存储器时，分时提供低 8 位地址和 8 位双向数据。在对单片机内部 EPROM 进行编程和校验时，P0 端口用于数据的输入和输出。

（2） P1 端口（P1.0～P1.7）

P1 端口是一个内部带有上拉电阻器的 8 位准双向 I/O 端口。

（3） P2 端口（P2.0～P2.7）

P2 端口的第一功能是作为一个内部带有上拉电阻器的 8 位准双向 I/O 端口。第二功能是在访问外部存储器时，输出高 8 位地址。

（4） P3 端口（P3.0～P3.7）

P3 端口的第一功能是作为一个内部带有上拉电阻器的 8 位准双向 I/O 端口。在系统中，这 8 个引脚又具有各自的第二功能，见表 1-1 所列。

表 1-1　P3 端口的第二功能

引　脚	第二功能名称	第二功能说明
P3.0	RXD0	串行数据输入端
P3.1	TXD0	串行数据输出端
P3.2	$\overline{\text{INT0}}$	外部中断 0 输入端
P3.3	$\overline{\text{INT1}}$	外部中断 1 输入端
P3.4	T0	定时/计数器 T0 外部输入端
P3.5	T1	定时/计数器 T1 外部输入端
P3.6	$\overline{\text{WR}}$	外部数据存储器写选通信号端
P3.7	$\overline{\text{RD}}$	外部数据存储器读选通信号端

温馨提示：

P0 端口的每一位能驱动 8 个 LSTTL 门的输入端，P1～P3 端口的每一位能驱动 3 个 LSTTL 门的输入端。

1.5.3 MCS—51 系列单片机的基本结构

MCS—51 系列单片机主要有 8031、8051 和 8751 三种基本产品，它们的内部结构基本相同，唯一的区别是内部程序存储器 ROM 配置不一样。8031 内部没有 ROM，8051 内部含有 4KB 的掩模 ROM，8751 内部含有 4KB 的 EPROM。下面以 8051 为例介绍 MCS—51 系列单片机的基本结构。

8051 型单片机结构图如图 1-5 所示。8051 型单片机主要包括以下几个部分。

- 1 个 8 位的 CPU。
- 1 个内部时钟电路。
- 4 KB 的掩模 ROM。
- 128 B 的数据存储器 RAM。
- 可寻址外部程序存储器和数据存储器空间为 64 KB。
- 21 个特殊功能寄存器 SFR。
- 4 个 8 位的并行 I/O 端口。
- 1 个全双工串行口。
- 2 个 16 位的定时/计数器。
- 5 个中断源，两级中断优先级。

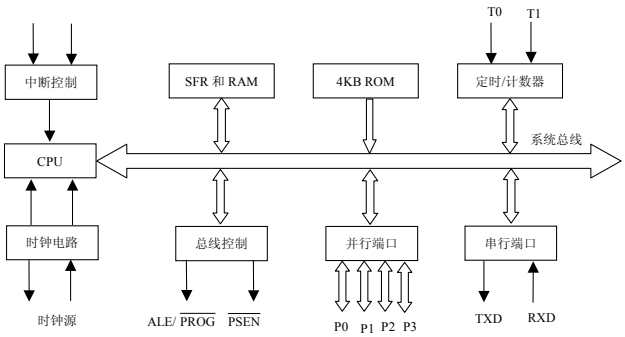


图 1-5　8051 型单片机结构图

1.5.4　MCS—51 系列单片机时钟电路与 CPU 时序

1.　时钟电路

单片机的时钟电路可以分为内部时钟电路方式和外部时钟电路方式两种。

（1）　内部时钟电路方式

内部时钟电路方式如图 1-6 所示。内部时钟电路方式由单片机外接晶体振荡器 Y、电容器 C1 和 C2 以及单片机内部振荡电路组成。晶体振荡器的频率为 0～40 MHz，典型值为 6 MHz 和 12 MHz，补偿电容器 C1 和 C2 的值为 5～30 pF，典型值为 30 pF。

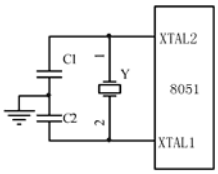


图 1-6　内部时钟电路方式

（2）　外部时钟电路方式

外部时钟电路方式如图 1-7 所示。从图中可以看出，时钟信号是由外部引入的。对于 TTL 型 CPU 采用方式 1，对于 CMOS 型 CPU 采用方式 2。

（a）方式 1　　（b）方式 2

图 1-7　外部时钟电路方式

2.　CPU 时序

单片机执行指令时总是按照时钟节拍自动地、一步一步地执行，各指令操作有严格的时间顺序，这种时间顺序被称为 CPU 的时序。CPU 时序如图 1-8 所示。为了更好地理解它的时序，下面介绍几个与它有关的概念。

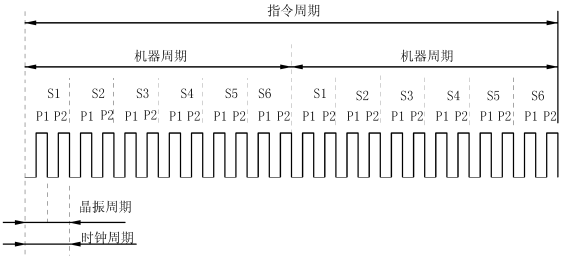


图 1-8　CPU 时序

（1）　晶体振荡器周期

晶体振荡器周期是最小的时序单位，它是晶体振荡频率（f_{osc}）的倒数。例如晶体振荡器频率为 6 MHz，则晶体振荡器周期为 167ns。

（2）　时钟周期

时钟周期又称为状态周期，从图 1-8 中可以看出，时钟周期是晶体振荡周期的两倍，可以分为 P1 和 P2 两个节拍，一般 P1 节拍完成算术逻辑操作，P2 节拍完成内部寄存器之间的数据传送操作。

（3） 机器周期

1 个机器周期为 12 个晶体振荡周期，一般来说，指令的执行时间通常用机器周期作为基本单位来衡量。

（4） 指令周期

指令周期是单片机执行一条指令所用的时间，MCS—51 系列单片机的指令按指令执行时间的长短可以分为单周期（即 1 个机器周期）指令、双周期指令和四周期指令 3 种，其中只有乘法和除法指令为四周期指令，其余指令均为单周期指令和双周期指令。

1.5.5 MCS—51 系列单片机的复位及复位电路

1. 复位

单片机的复位与计算机的重启类似，其目的是让单片机处于一个确定的初始状态。只要在单片机的 RST 引脚上保持两个以上机器周期的高电平，单片机就可以实现复位。复位后单片机从 0000H 单元开始执行指令。

2. 复位电路

常用的复位电路有自动复位和手动复位两种。

（1） 自动复位电路

自动复位又称上电复位，图 1-9 所示为自动复位电路，由电容器 C 和电阻器 R 组成。接通电源瞬间，电源经过电容器 C 加到 RST 引脚，使电路实现复位。

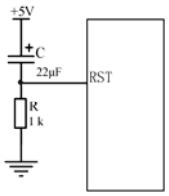


图 1-9 自动复位电路

（2） 手动复位电路

手动复位电路如图 1-10 所示，它由复位开关 S1、电阻器 R1 和 R2 组成。当按下复位开关 S1 时，+5 V 经 S1、R1 加到 RST 引脚，使电路实现复位。

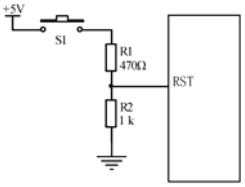


图 1-10　手动复位电路

1.5.6　MCS—51 系列单片机的存储器结构

存储器负责单片机的记忆工作，用于存储程序、常数、原始数据、中间结果和最终结果等，在应用单片机时，经常要使用到单片机的存储器，下面将以 8051 型单片机为例介绍 MCS—51 系列单片机的存储器结构。

单片机的存储器分为程序存储器（ROM）和数据存储器（RAM）两大类，8051 型单片机的存储器结构如图 1-11 所示。

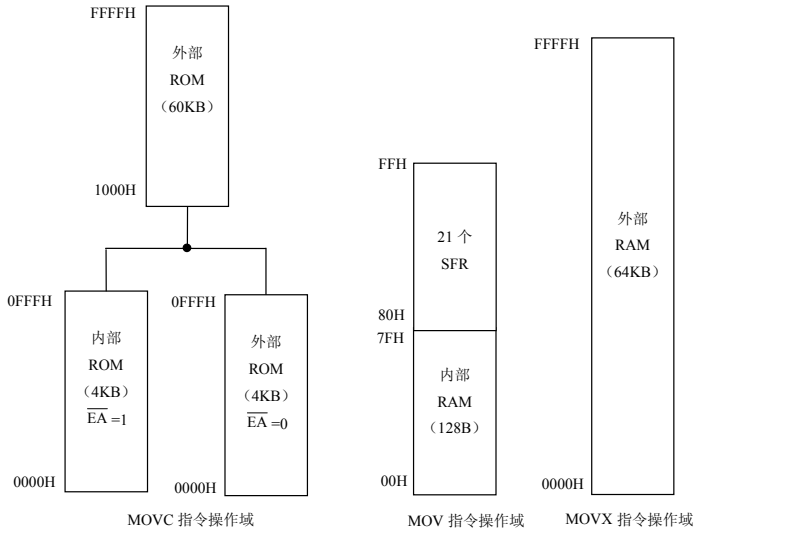


图 1-11　8051 型单片机的存储器结构

1.　程序存储器（ROM）

程序存储器用于存放固定程序和常数，当关闭电源时，其所存储的信息不会丢失。8051 型单片机内部含有 4 KB 程序存储器（ROM），地址范围为 0000H～0FFFH。另外，它还可以在外部扩展 60 KB 的外部程序存储器，地址范围为 1000H～FFFFH，这两者可以统一编址。

温馨提示：

在程序存储器中，有些单元具有特殊含义，使用时必须注意以下几点。

- 0000H：单片机复位后的入口地址。PC=0000H，程序从 0000H 开始执行指令；
- 0003H：$\overline{\text{INT0}}$ 入口地址；
- 000BH：定时/计数器 T0 溢出中断入口地址；
- 0013H：$\overline{\text{INT1}}$ 入口地址；
- 001BH：定时/计数器 T1 溢出中断入口地址；
- 0023H：串口中断入口地址。

2. 数据存储器（RAM）

数据存储器用于存放单片机运行时临时产生的中间结果，当关闭电源时，其所存储的信息将会丢失。单片机的数据存储器可以分为内部数据存储器和外部数据存储器两类。

（1） 内部数据存储器

单片机的内部数据存储器由低 128B 数据存储器（00H～7FH）和 21 个特殊功能寄存器（SFR）两部分组成。

①低 128B 数据存储器（00H～7FH）。低 128B 数据存储器是单片机的真正数据存储器，其结构如图 1-12 所示。由图可知，该数据存储器按用途可以分为工作寄存器区、位寻址区和通用 RAM 区。

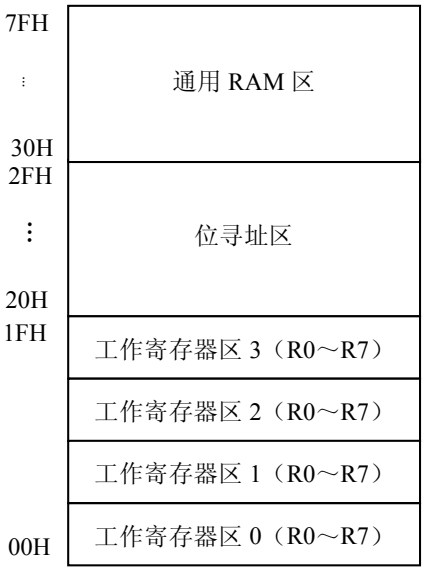


图 1-12　低 128B 数据存储器的结构

- 工作寄存器区（00H～1FH）

8051 型单片机共有 4 组工作寄存器，编号依次为工作寄存器 0、1、2 和 3，每组工作寄存器有 8 个单元，均以 R0～R7 命名。通过对程序状态字 PSW 中的 RS0 和 RS1 位设置，可以选择不同的工作寄存器区。工作寄存器区的设置见表 1-2 所列。

表 1-2　工作寄存器区的设置

设 置 值		选择工作寄存器区	工作寄存器单元							
RS1	RS0		R0	R1	R2	R3	R4	R5	R6	R7

0	0	0	00H	01H	02H	03H	04H	05H	06H	07H
0	1	1	08H	09H	0AH	0BH	0CH	0DH	0EH	0FH
1	0	2	10H	11H	12H	13H	14H	15H	16H	17H
1	1	3	18H	19H	1AH	1BH	1CH	1DH	1EH	1FH

温馨提示：

系统复位或上电后，自动选择工作寄存器 0。

➢ 位寻址区

内部数据存储器的 20H～2FH 单元，既可作为一般 RAM 使用，进行字节操作，也可以对单元的每一位进行位操作，因此把该区称为位寻址区，内部数据存储器位寻址区的位地址见表 1-3 所列。

表 1-3　内部数据存储器位寻址区的位地址

字节地址	位 地 址							
	D7	D6	D5	D4	D3	D2	D1	D0
2FH	7FH	7EH	7DH	7CH	7BH	7AH	79H	78H
2EH	77H	76H	75H	74H	73H	72H	71H	70H
2DH	6FH	6EH	6DH	6CH	6BH	6AH	69H	68H
2CH	67H	66H	65H	64H	63H	62H	61H	60H
2BH	5FH	5EH	5DH	5CH	5BH	5AH	59H	58H
2AH	57H	56H	55H	54H	53H	52H	51H	50H
29H	4FH	4EH	4DH	4CH	4BH	4AH	49H	48H
28H	47H	46H	45H	44H	43H	42H	41H	40H
27H	3FH	3EH	3DH	3CH	3BH	3AH	39H	38H
26H	37H	36H	35H	34H	33H	32H	31H	30H
25H	2FH	2EH	2DH	2CH	2BH	2AH	29H	28H
24H	27H	26H	25H	24H	23H	22H	21H	20H
23H	1FH	1EH	1DH	1CH	1BH	1AH	19H	18H
22H	17H	16H	15H	14H	13H	12H	11H	10H
21H	0FH	0EH	0DH	0CH	0BH	0AH	09H	08H
20H	07H	06H	05H	04H	03H	02H	01H	00H

➢ 通用 RAM 区

内部数据存储器的 30H～7FH 字节单元为通用 RAM 区，也称用户 RAM 区，共 80B，可以作为一般数据存储区和堆栈区使用。

② 21 个特殊功能寄存器（SFR）。8051 型单片机共有 21 个特殊功能寄存器，简记为 SFR（Special Function Registers），它们离散地分布在 80H～FFH 地址范围内。表 1-4 所列为 8051 型单片机特殊功能寄存器的分布。

表 1-4　8051 型单片机特殊功能寄存器的分布

SFR 名称	地址	位地址/位名称								复 位 值
		MSB							LSB	
B*	F0H	F7	F6	F5	F4	F3	F2	F1	F0	00H
		B.7	B.6	B.5	B.4	B.3	B.2	B.1	B.0	

A_{CC}*	E0H	E7	E6	E5	E4	E3	E2	E1	E0	00H
		A_{CC}.7	A_{CC}.6	A_{CC}.5	A_{CC}.4	A_{CC}.3	A_{CC}.2	A_{CC}.1	A_{CC}.0	
PSW*	D0H	D7	D6	D5	D4	D3	D2	D1	D0	000000x0B
		Cy	AC	F0	RS1	RS0	OV		P	
IP	B8H	BF	BE	BD	BC	BB	BA	B9	B8	xxx00000B
					PS	PT1	PX1	PT0	PX0	

（续表）

SFR 名称	地址	位地址/位名称 MSB							LSB	复位值
P3*	B0H	B7	B6	B5	B4	B3	B2	B1	B0	FFH
		P3.7	P3.6	P3.5	P3.4	P3.3	P3.2	P3.1	P3.0	
IE*	A8H	AF	AE	AD	AC	AB	AA	A9	A8	0x000000B
		EA			ES	ST1	EX1	ET0	EX0	
P2*	A0H	A7	A6	A5	A4	A3	A2	A1	A0	FFH
		P2.7	P2.6	P2.5	P2.4	P2.3	P2.2	P2.1	P2.0	
SBUF	99H									xxxxxxxxB
SCON*	98H	9F	9E	9D	9C	9B	9A	99	98	00H
		SM0	SM1	SM2	REN	TB8	RB8	TI	RI	
P1*	90H	97	96	95	94	93	92	91	90	FFH
		P1.7	P1.6	P1.5	P1.4	P1.3	P1.2	P1.1	P1.0	
TH1	8DH									
TH0	8CH									
TL1	8BH									
TL0	8AH									
TMOD	89H	GATE	C/T	M1	M0	GATE	C/T	M1	M0	00H
TCON*	88H	8F	8E	8D	8C	8B	8A	89	88	00H
		TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	
PCON	87H	SMOD				GF1	GF0	PD	IDL	xxxxxxxxB
DPH	83H									00H
DPL	82H									00H
SP	81H									07H
P0*	80H	87	86	85	84	83	82	81	80	FFH
		P0.7	P0.6	P0.5	P0.4	P0.3	P0.2	P0.1	P0.0	

温馨提示：

凡字节地址能被 8 整除（即十六进制地址码尾数为 8 或 0 的地址），可以位寻址，表 1-4 中用“*”表示。

下面简单介绍部分特殊功能寄存器，其他没有介绍的将在后续相关章节中介绍。

➢ 累加器 A_{CC}

累加器 A_{CC} 是一个 8 位寄存器，通常简称 A，是最常用的专用寄存器，它既可以存放指令的操作数，也可以存放运算结果。

➢ B 寄存器

B 寄存器是一个 8 位寄存器，主要用于乘除运算。做乘法运算时，B 存放乘数，乘法操作后，结果的高 8 位存放在 B 寄存器中。做除法运算时，B 存放除数，除法运算后，余数存放在 B 中。

➢ 程序状态字 PSW

程序状态字 PSW 是一个 8 位寄存器，主要用于存放程序运行中的各种状态信息。其中部分位的状态是根据程序执行的结构由硬件自动设置的，部分位是由用户设置的。程序状态字的各位定义见表 1-5 所列。

表 1-5　程序状态字各位的定义

位　序	D7	D6	D5	D4	D3	D2	D1	D0
位 标 志	Cy	AC	F0	RS1	RS0	OV	-	P

Cy：借/进位标志。在执行减法或加法时，若结果的最高位产生进位或借位，Cy 由硬件自动置 1，否则清零。

AC：辅助借/进位标志。在执行减法或加法时，若结果的低 4 位向高 4 位（即 D3 位）产生进位或借位，AC 由硬件自动置 1，否则清零。

F0：用户标志。由用户根据需要对 F0 置 1 或清零，作为软件标志。

RS1、RS0：工作寄存器组选择控制位。由用户根据需要设置 RS1 和 RS0 的值，从而实现选择不同的工作寄存器组。表 1-6 所列为当前工作寄存器的选择。

表 1-6　当前工作寄存器的选择

RS1	RS0	选择个工作寄存器组（R0～R7）
0	0	工作寄存器组 0
0	1	工作寄存器组 1
1	0	工作寄存器组 2
1	1	工作寄存器组 3

OV：溢出标志。在执行算术运算时，如果结果的最高位（即 D7 位）和次高位（即 D6 位）中有且只有一位产生借/进位，则 OV 置 1，说明运算结果超出有效范围（-128～+127），否则清零。

P：奇偶标志。若累加器 A 中的 1 个数为奇数，则 P 置 1，否则清零。

例如：累加器 A=97H，执行加法指令“ADD A，#79H”，情况如下所示。

```
    D7  D6  D5  D4  D3  D2  D1  D0
     1   0   0   1   0   1   1   1   B ；97H
+1   0   1   1   1   1   0   0   1   B ；79H
-------------------------------------
     0   0   0   1   0   0   0   0   B ；10H
```

D7 产生进位，则 Cy =1。

D3 向高 4 位有进位，则 AC=1。

D7 和 D6 同时向高 1 位有进位，则 OV=0。

累加器 A 有一个 1，为奇数，则 P=1。

➢ 堆栈指针 SP

堆栈指针 SP（Stack pointer）是一个 8 位的寄存器，堆栈主要用于响应中断或调用子程序时，保护断点地址。单片机复位后，SP 为 07H。

温馨提示：

中断时，程序断点地址会自动压入堆栈。具体过程是数据进入堆栈前，SP 先自动加 1，然后将低 8 位地址压入堆栈，SP 再加 1 后，再将高 8 位地址压入堆栈。

➢ 数据指针 DPTR

数据指针 DPTR（Data pointer）是一个 16 位的寄存器，它由高 8 位寄存器 DPH 和低 8 位寄存器 DPL 组成，主要用于存放 16 位地址。

（2） 外部数据存储器

8051 型单片机根据需要，可以扩展外部数据存储器，地址范围为 0000H～FFFFH，共 64KB，可通过 MOVX 指令访问。

1.6　如何设计 LED 发光二极管与单片机接口电路

LED 发光二极管与单片机接口电路如图 1-13 所示。图中 U1（AT89C51）是一个 51 系列的单片机，内部含有 4KB 电可擦除的程序存储器（简称 E^2PROM）；U2（74LS245）为驱动器，为发光二极管提供足够大的驱动电流；开关 S1、电容器 C3 以及电阻器 R2 和 R3 共同组成单片机的复位电路；晶体振荡器 Y、电容器 C1 和 C2 组成时钟电路；发光二极管 VD1、限流电阻器 R1 组成显示电路。

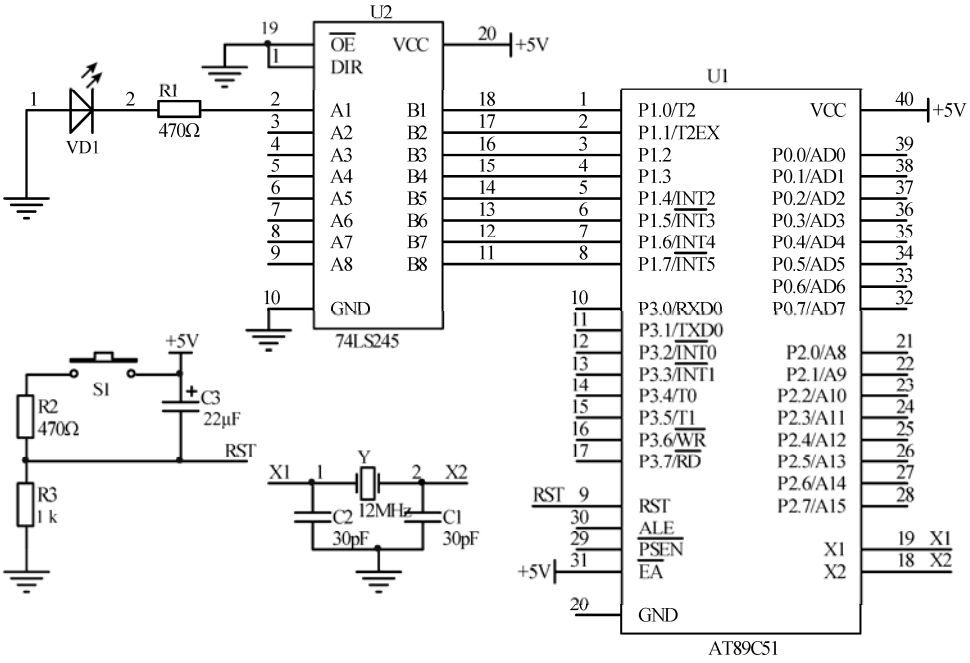


图 1-13　LED 发光二极管与单片机接口电路

当单片机的 P1.0 端输出高电平 1 时，U2 的 $\overline{OE}$ 端为 1，经过 U2 驱动后从其 A1 端输出 1，该高电平经限流电阻器 R1 加到发光二极 VD1 的负极，VD1 熄灭。同理，当单片机

的 P1.0 端输出低电平 0 时，U2 的$\overline{OE}$端为 0，经过 U2 驱动后从其 A1 端输出 0，该低电平经限流电阻器 R1 后加到发光二极管 VD1 的负极，VD1 亮。

1.7 如何设计单个彩灯闪烁程序

1.7.1 置 1 和清零指令

要实现 LED 发光二极管的亮/灭，P1.0 端必须输出高电平 1 或低电平 0，下面介绍置 1 和清零指令。

1. SETB 指令

SETB 指令的作用是置 1，指令格式有以下两种。

（1） SETB bit

SETB bit 指令的作用是将 bit 位置 1。

（2） SETB C

SETB C 指令的作用是将借/进位标志置 1。例如 SETB P1.0，其作用是将 P1.0 端置 1。

2. CLR 指令

CLR 指令的作用是清零，指令格式有以下两种

（1） CLR bit

CLR bit 指令的作用是将 bit 位清零。

（2） CLR C

CLR C 指令的作用是将借/进位标志清零。例如 CLR P1.0，其作用是将 P1.0 端清零。

1.7.2 延时子程序

由于人眼的视觉惰性，在单个彩灯闪烁程序中必须调用延时子程序，让 LED 发光二极管的亮和灭均保持一段时间。延时子程序是通过循环执行某些指令实现的，下面将具体介绍延时子程序的编写方法。

1. MOV、DJNZ 和 RET 指令

在延时子程序中将用到 MOV、DJNZ 和 RET 三条指令，下面先简单介绍这三条指令。

（1） MOV 指令

MOV 指令的作用是数据传送，例如：

```
MOV  R1，#200
```

执行上述指令后，（R1）=200

（2） DJNZ 指令

DJNZ 指令的作用是减 1 不为 0 跳转，例如：

```
            MOV  R2，#250
LOOP2：     DJNZ  R2，LOOP2
            MOV  R3，#30
```

在本例中，要执行 250 次 DJNZ　R2，LOOP2 指令后（即 R2 中内容为 0），才执行 MOV　R3，#30 指令。

（3） RET 指令

RET 指令的作用是子程序返回，即返回到调用该子程序处。

2. 延时子程序

若晶体振荡器的频率为 12 MHz，则一个机器周期为 1 μs，延时子程序如下所示。

```
DEALAY：    MOV  R1，#200        ；1 μs
LOOP1：     MOV  R2，#250        ；1 μs
LOOP2：     DJNZ  R2，LOOP2      ；2 μs，R2-1 不为 0 转 LOOP2
            DJNZ  R1，LOOP1      ；2 μs，R1-1 不为 0 转 LOOP1
            RET                  ；2 μs，子程序返回
```

上述延时子程序采用了两级循环。在内循环中，循环次数放在寄存器 R2 中，共执行内循环 DJNZ　R2，LOOP2 指令 250 次，每次执行时间为 2 μs，累计 250×2 μs = 500 μs。在外循环中，循环次数放在 R1 中，执行 1 次大循环的时间为 1 μs（执行 MOV R2，#250 指令时间）+ 500 μs（内循环时间）+ 2 μs（执行 DJNZ　R1，LOOP1 指令时间）= 503 μs。则总延时时间为 1 μs（执行 MOV R1，#200 指令时间）+200×503 μs=100601 μs = 100.6 ms，约为 101 ms。

1.7.3　子程序调用和长跳转指令

1. 子程序调用指令

由于单个彩灯闪烁程序内部有子程序，下面介绍子程序调用指令 CALL，指令格式如下所示。

```
CALL 子程序名
```

例如 CALL DELAY 指令的作用是调用 DELAY 子程序。

2. 长跳转指令

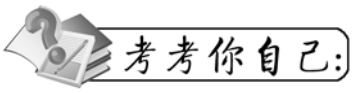
LJMP 指令为长跳转指令，其作用是跳转到指定地址或标号去执行程序，指令格式如下所示。

```
LJMP  addr16 或标号
```

例如 LJMP START 指令的作用是跳转去 START 处执行。

1.7.4 单个彩灯程序

```
        ORG  0000H
START:  SETB  P1.0
        CALL  DELAY
        CLR  P1.0
        CALL  DELAY
        LJMP  START
DEALAY: MOV  R1,#200        ; 1 μs
LOOP1:  MOV  R2,#250        ; 1 μs
LOOP2:  DJNZ  R2,LOOP2      ; 2 μs，R2-1 不为 0 转 LOOP2
        DJNZ  R1,LOOP1      ; 2 μs，R1-1 不为 0 转 LOOP1
        RET
```



1. 发光二极管具有什么特性？与普通二极管有何异同？
2. CPU 的 $\overline{EA}$ 引脚有何作用？
3. 把下列十进制数转换成二进制数和十六进制数。
   （1）35　（2）0.25　（3）95　（4）17　（5）29.75
4. 把下列二进制数转换成十进制数和十六进制数。
   （1）1011B　（2）11.1011B　（3）1101.101B　（4）1101101.11B
5. 把下列十六进制数转换成二进制数和十进制数。
   （1）15H　（2）4AH　（3）BBH　（4）13.5H　（5）27.AH
6. 若延时时间为 0.8 s，则应对程序做哪些修改？

项目二　广告灯控制

——输入/输出端口应用

教学目的

掌握：发光二极管与单片机接口电路设计方法。
理解：数据传送类指令和控制转移类指令的应用。
了解：寻址方式。

愿你知多点：

在日常生活中，我们经常看到各种类型的广告灯，例如超级市场门前的广告灯、城市休憩广场的装饰灯、公路两旁的霓虹灯等，那么这些广告灯是如何制作和控制的呢？在这一章中，我们将通过完成广告灯控制这一任务来学习发光二极管的相关知识及制作广告灯的方法。图 2-1 所示为几个广告灯的应用实例。

图 2-1　几个广告灯的应用实例

2.1　能力培养

本章通过完成广告灯控制这一任务，可以培养以下能力。

1. 能正确使用发光二极管。
2. 能计算指令执行时间。
3. 能正确实现对广告灯的控制。

2.2　任务分析

要完成此项任务，需要掌握以下 6 方面知识。

1. 寻址方式。
2. 如何使用数据传送类指令。
3. 如何使用控制转移类指令。
4. 如何计算指令执行时间。
5. 如何设计发光二极管与单片机接口电路。
6. 如何设计广告灯程序。

2.3 寻址方式

操作数是指令的一个重要组成部分，CPU 在规定的寻址空间获得操作数地址的方式，称为寻址方式。寻址方式不仅影响指令的长度，还影响指令的执行速度。

MSC—51 系列单片机的寻址方式有立即寻址、直接寻址、寄存器寻址、寄存器间接寻址、变址寻址、相对寻址和位寻址七种。

2.3.1 立即寻址方式

立即寻址方式示意图如图 2-2 所示。操作数作为指令的一个组成部分存放在程序存储器中，这种寻址方式称为立即寻址，该操作数称为立即数。立即数前加“＃”标记，以便和直接寻址方式相区分。

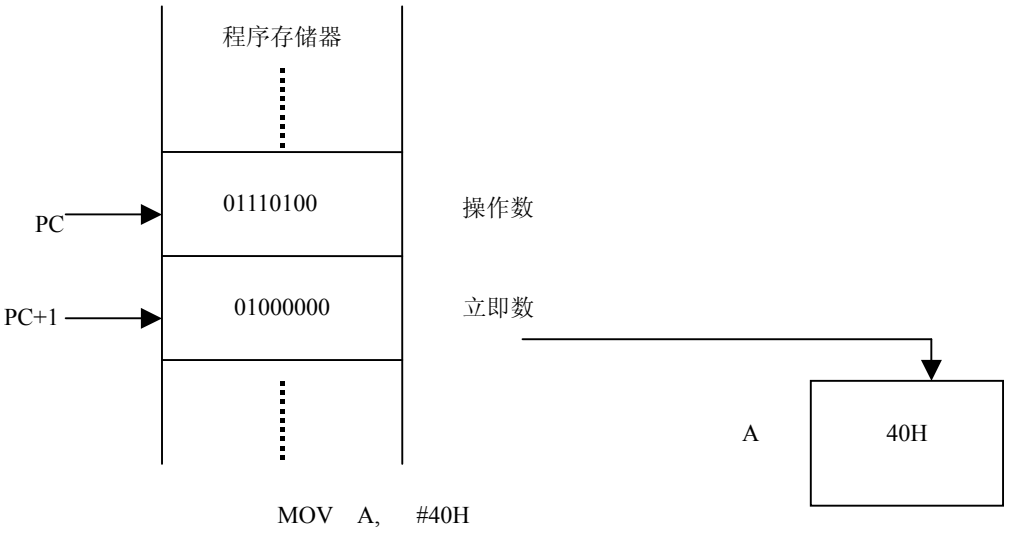


图 2-2 立即寻址方式示意图

例如指令 `MOV A，＃40H ；A←40H`

该指令将立即数 40H 送至累加器 A 中。指令执行后，A=40H，指令中用“＃”符号表示立即数。

2.3.2　直接寻址方式

直接寻址方式是在指令代码中直接给出操作数的地址。这种寻址方式是对内部数据存储器进行访问。

例如指令：　MOV　A，50H　　；A←（50H）

该指令是把内部 RAM 中 50H 单元（直接寻址）的内容送至累加器 A 中。假设指令执行前 A 的内容为 8BH（表示为 A=8BH），50H 单元的内容为 36H（表示为（50H）=36H），则指令执行后，A=36H，（50H）=36H（不变）。直接寻址方式示意图如图 2-3 所示。

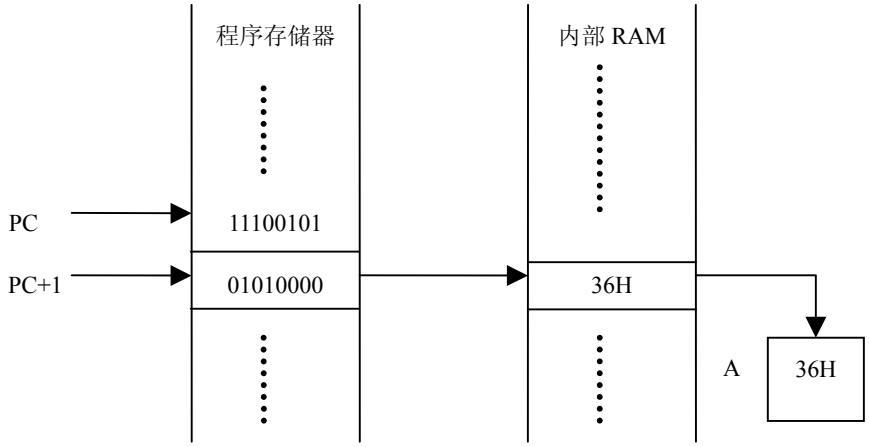


图 2-3　直接寻址方式示意图

温馨提示：

直接寻址方式可寻址的空间如下所示。

1. 内部 RAM 的低 128 字节单元地址空间。
2 特殊功能寄存器 SFR 地址空间（直接寻址方式是访问 SFR 的唯一方式）。
3 位地址空间。

2.3.3　寄存器寻址方式

寄存器寻址方式是指，指令中给出寄存器名称，其内容作为操作数。这种寻址方式对当前工作寄存器 R0 ～ R7 进行访问，指令操作码的低 3 位指明了所用的寄存器；对累加器 A、寄存器 B 及数据指针 DPTR 也可以用寄存器寻址方式来访问。

例如指令：　MOV　A，　R2　　；A←（R2）

该指令是将工作寄存器 R2 中的内容送至累加器 A 中。假设指令执行前 A 的内容为 08H（表示为 A=08H），R2 的内容为 4EH（表示为 R2=4EH），则指令执行后，A=4EH，R2=4EH

（不变）。寄存器寻址方式示意图如图 2-4 所示。

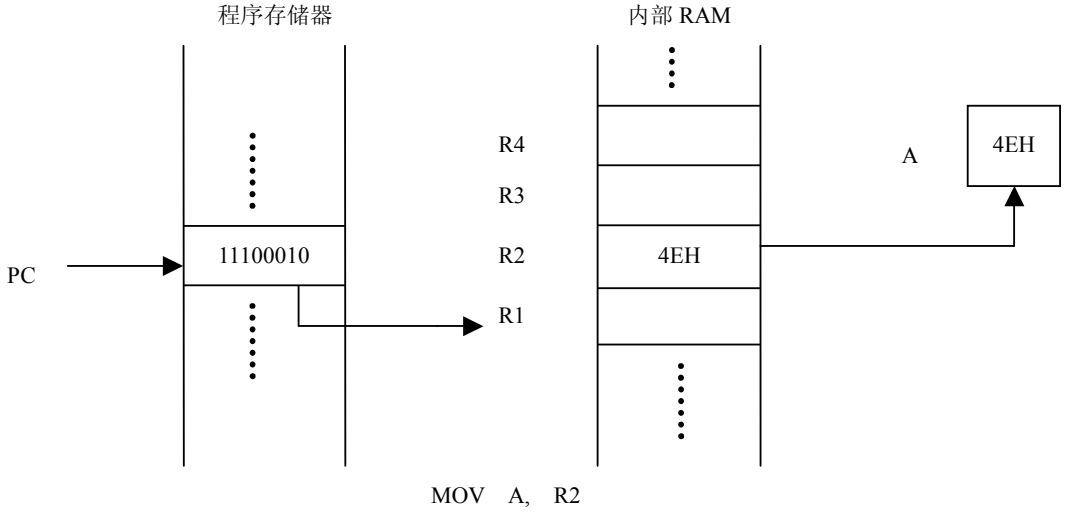


图 2-4　寄存器寻址方式示意图

温馨提示：

寄存器寻址方式的可寻址空间如下所示。

1. 当前工作寄存器 R0 ～ R7（当前工作寄存器区由 RS1、RS2 来选定）。
2. 累加器 A、寄存器 B 及数据寄存器 DPTR。

2.3.4　寄存器间接寻址方式

指令中给出寄存器，以寄存器的内容作为操作数的地址，把该地址对应单元的内容作为操作数。

在访问外部 RAM 整个 256 个单元（××00H ～ ××FFH）时，用 R0，R1 工作寄存器间接寻址。

在访问外部 RAM 整个 64KB（0000H ～ FFFFH）地址空间时，用数据指针 DPTR 来间接寻址。

例如指令：`MOV A, @R0　　；A←（R0）`

该指令的操作为将寄存器 R0 的内容（设（R0）=50H）作为地址，把内部 RAM 的 50H 单元的内容（设（50H）=48H）送至累加器 A 中。指令执行后，A=48H。指令中“@”表示寄存器间接寻址，则称之为间址符。寄存器间接寻址方式示意如图 2-5 所示。

温馨提示：

寄存器间接寻址方式的可寻址空间如下所示。

1. 内部 RAM：00H～7FH（@R0、@R1、SP）。
2. 外部 RAM：0000H～FFFFH（@R0、@R1、@DPTR）。

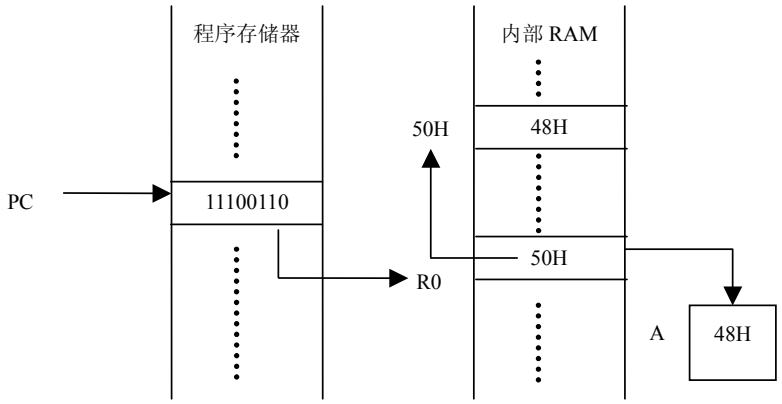


图 2-5　寄存器间接寻址方式示意图

2.3.5　变址寻址方式

变址寻址方式是 MCS—51 系列单片机的指令系统中所特有的一种寻址方式。它以程序计数器 PC 或数据指针 DPTR 作为基址寄存器，以累加器 A 作为变址寄存器（存放 8 位无符号的偏移量），两者内容相加形成 16 位程序存储器地址作为指令操作数的地址。这种寻址方式用于读取程序存储器中的常数表。

例如指令：`MOVC　A，　@A+DPTR　　；A←A+DPTR`

该指令是把 DPTR 的内容作为基地址，把累加器 A 中的内容作为偏移量，两量相加形成 16 位地址，将该地址的程序存储器 ROM 单元的内容送至 A 中。假设指令执行前 DPTR=2100H，A=56H，（2156H）=36H，则该指令执行后，A=36H。变址寻址方式示意图如图 2-6 所示。

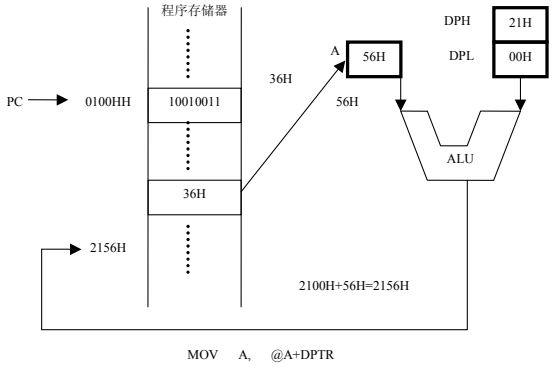


图 2-6　变址寻址方式示意图

温馨提示：

变址寻址方式的可寻址空间为程序存储器（@A+DPTR，@A+PC）

2.3.6 相对寻址方式

相对寻址方式示意图如图 2-7 所示，只用于相对转移指令中。相对转移指令是以本指令的下一条指令的首地址 PC 为基地址，与指令中给定的相对偏移量 rel 相加之和作为程序的转移目标地址。偏移量 rel 是 8 位二进制补码（与 PC 相加时，rel 需符号扩展成 16 位）。转移范围为当前 PC 值的−128～ +127 个字节单元之间。相对寻址指令一般为双字节或三字节指令。

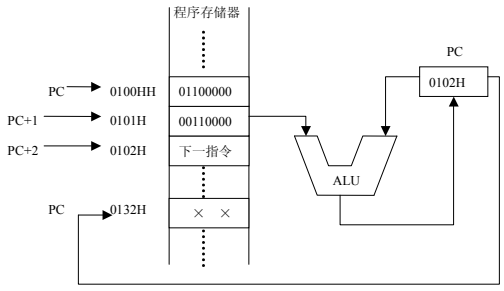


图 2-7 相对寻址方式示意图

例如指令： JZ 30H ；当 A=0 时，则 PC←PC+2+REL，程序转移。
；当 A=1 时，则 PC←PC+2。程序按原顺序执行。

温馨提示：

相对寻址方式的可寻址空间为程序存储器。

2.3.7 位寻址方式

MCS—51 系列单片机中设有独立的位处理器。位操作命令能对可以位寻址的空间进行位操作。

例如指令： MOV C，07H ；Cy ←（07H）

该指令属位操作指令，将内部 RAM 的 20H 单元的 D7 位（位地址为 07H）的内容送给位累加器 Cy，指令的操作过程如图 2-8 所示。

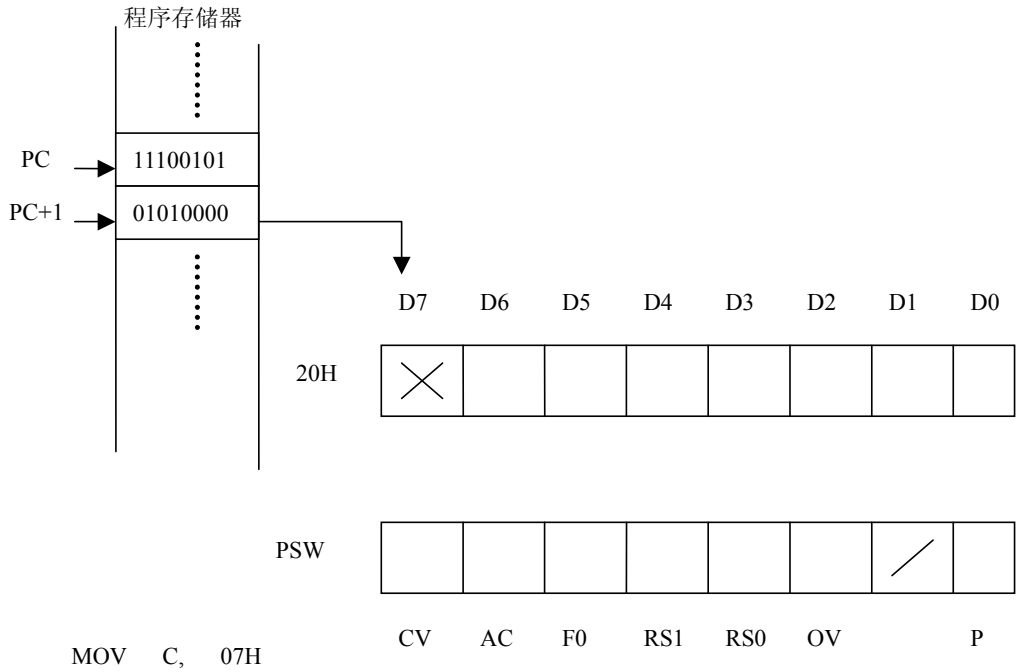


图 2-8　位寻址方式示意图

温馨提示：

位寻址方式的可寻址空间如下所示。

1. 内部 RAM 的位寻址区域为 20H～2FH，共 16 个单元 128 位，其位地址编码为 00H～7FH。

2. 字节地址能被 8 整除的 SFR（12 个）。对这些寻址位，可以有以下 4 种表示方法。

（1） 直接位地址方式，如 0D5H。

（2） 位名称方式，如 F0。

（3） 点操作符方式，如 PSW · 5 或 0D0H · 5。

（4） 用户定义名方式，如用伪指令 bit 定义 USR_FLG　bit　F0，经定义后，允许指令用 USR_FLG 代替 F0。

以上 4 种方式指的都是 PSW 中的第 5 位。

2.4　如何使用数据传送类指令

数据传送类指令共有 29 条，分为内部寄存器、数据存储器传送指令（MOV）、外部数据存储器传送指令（MOVX）、程序存储器内查表指令（MOVC）、（XCH、XCHD、SWAP）数据交换指令和堆栈操作指令（PUSH、POP）。源操作数可以采用寄存器寻址、寄存器间接寻址、直接寻址、立即寻址和变址寻址 5 种方式，目的操作数可以采用前 3 种寻址方式。

数据传送类指令是编程时使用最频繁的一类指令。

数据传送类指令一般的操作是把源操作数传送到目的操作数，指令执行后，源操作数不改变，目的操作数修改为源操作数。如果要求在数据传送，不丢失目的操作数，则可以用交换指令。

数据传送类指令不影响标志。这里所说的标志是指 C，AC 和 OV，不包括检验累加器奇偶标志位 P。如果执行结果改变累加器 A 的值时，会影响奇偶标志位 P。

2.4.1 内部数据存储器及寄存器间的数据传送指令（16 条）

1. 以累加器 A 为目的操作数的指令（4 条）

以累加器 A 为目的操作数的数据传送指令见表 2-1 所列。

表 2-1 以累加器 A 为目的操作数的数据传送指令

助 记 符	功能说明	机器周期
MOV A，Rn	将 Rn 中内容（简称 Rn 值，下同）送至 A 中，A←Rn。	1
MOV A，direct	将 direct 单元中内容（简称 direct 值，下同）送至 A 中，A←（direct）。	1
MOV A，#data	将 8 位常数＃data 送至 A 中，A←data。	1
MOV A，@Ri	将 Ri 间址的地址单元中内容（简称 Ri 间址值，下同）送至 A 中，A←（Ri）。	1

说明：

① 指令的功能是把源操作数的内容送至累加器 A 中。源操作数可以为寄存器寻址方式、直接寻址方式、寄存器间接寻址方式或立即寻址方式。

② 源操作数是 Rn 时，属寄存器寻址。MCS—51 系列单片机内部 RAM 区中有 4 组工作寄存器，每组由 8 个寄存器组成。可通过改变 PSW 中的 RS0，RS1 来切换当前工作寄存器组。

例 2-1

```
MOV  A，R4      ；A←R4
MOV  A，70H     ；A←（70H）
MOV  A，@R1     ；A←（R1）
MOV  A，＃88H   ；A←88H
```

2. 以 Rn 为目的操作数的指令（3 条）

以 Rn 为目的操作数的数据传送指令见表 2-2 所列。

表 2-2 以 Rn 为目的操作数的数据传送指令

助 记 符	功能说明	机器周期
MOV Rn，direct	将 direct 值送至 Rn 中，Rn←（direct）。	2
MOV Rn，#data	将常数 data 送至 Rn 中，Rn←data。	1
MOV Rn，A	将 A 值送至 Rn 中，Rn←A。	1

该组指令的功能是把源操作数的内容送至当前工作寄存器区中的 R7～R0 中的某一个寄存器。源操作数可以为寄存器寻址方式、直接寻址方式或立即寻址方式。

例 2-2
```
MOV  R2，A        ；R2←A
MOV  R6，70H      ；R6←（70H）
MOV  R4，#40H     ；R4←40H
```

3. 以直接地址为目的操作数的指令（5 条）

以直接地址为目的操作数的数据传送指令见表 2-3 所列。

表 2-3　以直接地址为目的操作数的数据传送指令

助 记 符	功能说明	机器周期
MOV　direct，Rn	将 Rn 值送至 direct 中。direct←Rn。	2
MOV　direct，A	将 A 值送至 direct 中，direct←A。	1
MOV　direct，@Ri	将 Ri 间址值送至 direct 中，direct←（Ri）。	2
MOV　direct，#data	将常数#data 直接送至 direct 中，direct←data 。	2
MOV　direct，direct2	将 direct1 值送至 direct2 中，direct1←（direct2）。	2

该组指令的功能是把源操作数的内容送至直接地址所指的存储单元。源操作数可以为寄存器寻址方式、直接寻址方式、寄存器间接寻址方式或立即寻址方式。

例 2-3
```
MOV  P1，A      ；  P1←A
MOV  64H，R6    ；  64H←R6
MOV  10H，#98H  ；  10H←98H
MOV  20H，48H   ；  20H←（48H）
MOV  24H，@R0   ；  24H←（R0）
```

4. 以寄存器间接地址为目的操作数的指令（3 条）

以寄存器间接地址为目的操作数的数据传送指令见表 2-4 所列。

表 2-4　以寄存器间接地址为目的操作数的数据传送指令

助 记 符	功能说明	机器周期
MOV　@Ri，A	将 A 值送入 Ri 指示的地址单元中，（Ri）←A。	1
MOV　@Ri，direct	将 direct 值送入 Ri 指示的地址单元中，（Ri）←（direct）。	2
MOV　@Ri，#data	将常数#data 直接送入 Ri 指示的地址单元中，（Ri）←data 。	1

该组指令的功能是把源操作数的内容送入 R0 或 R1 寄存器间接寻址所确定的内部 RAM 单元中。源操作数可以为寄存器寻址方式、直接寻址方式或立即寻址方式。

例 2-4
```
MOV  @R0，20H    ；  R0←（20H）
MOV  @R1，A      ；  R1←A
MOV  @R0，#28H   ；  R0←28H
```

5. 16 位数据传送指令（1 条）

16 位数据传送指令见表 2-5 所列。

表 2-5　16 位数据传送指令

助 记 符	功能说明	机器周期
MOV DPTR，#DATA16	将常数 DATA16 直接送入 DPTR 中，DPTR←DATA16。	2

该条指令是整个指令系统中唯一的一条 16 位数据传送指令，通常用来设置地址指针，DPTR 由 DPH 和 DPL 组成。该指令传送时，把高 8 位立即数送入 DPH，低 8 位立即数送入 DPL。

MOV 指令用于寻址内部 RAM 和 SFR。MOV 指令对内部 RAM 和 SFR 的操作功能如图 2-9 所示。

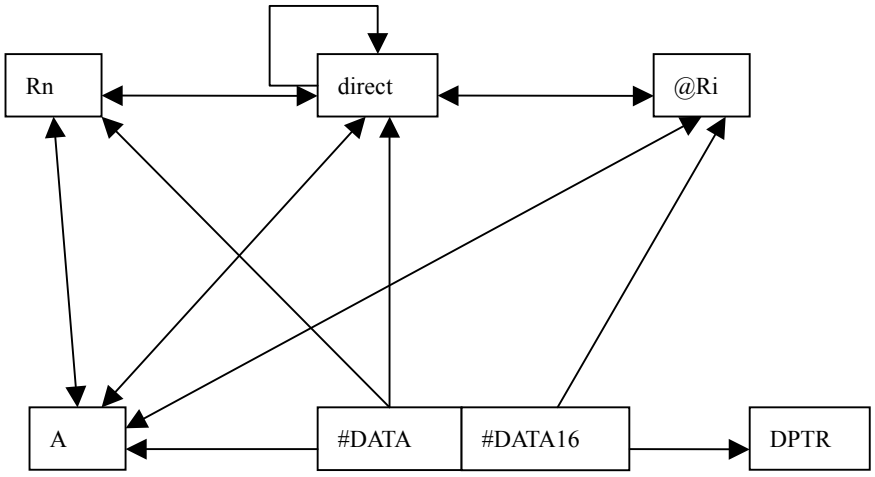


图 2-9　MOV 指令对内部 RAM 和 SFR 的操作功能

2.4.2　堆栈操作指令（2 条）

堆栈是在内部 RAM 中开辟的一个特定的存储区，栈顶地址由堆栈指针 SP 指示，SP 总是指向栈顶。堆栈按“先进后出”原则存取数据。堆栈操作指令有两种，数据存入称“入栈”或“压入（数据）”，数据取出称“出栈”或“弹出（数据）”。堆栈操作指令在子程序调用及中断服务时用于现场和返回保护，还可以用于参数传递等。

进栈指令先进行指针调整，即堆栈指针 SP 加 1，再把 direct 值入栈。出栈指令是先将栈顶内容弹出给 direct，再进行指针调整，即堆栈指针 SP 减“1”。表 2-6 所列为堆栈指令。

表 2-6　堆栈指令

助 记 符	功能说明	机器周期
PUSH　direct	入栈指令，将 direct 值压入堆栈栈顶，sp←sp+1，sp←（direct）。	2
POP　direct	出栈指令，将堆栈栈顶内容弹出到 direct 中，direct←(sp)，sp←sp-1。	2

例 2-5　已知（SP）=38H，（10H）=70H

执行指令：`PUSH　　10H`

指令执行后：（SP）=39H，（10H）=70H，（39H）=70H。
上述程序的指令入栈操作示意图如图 2-10 所示。

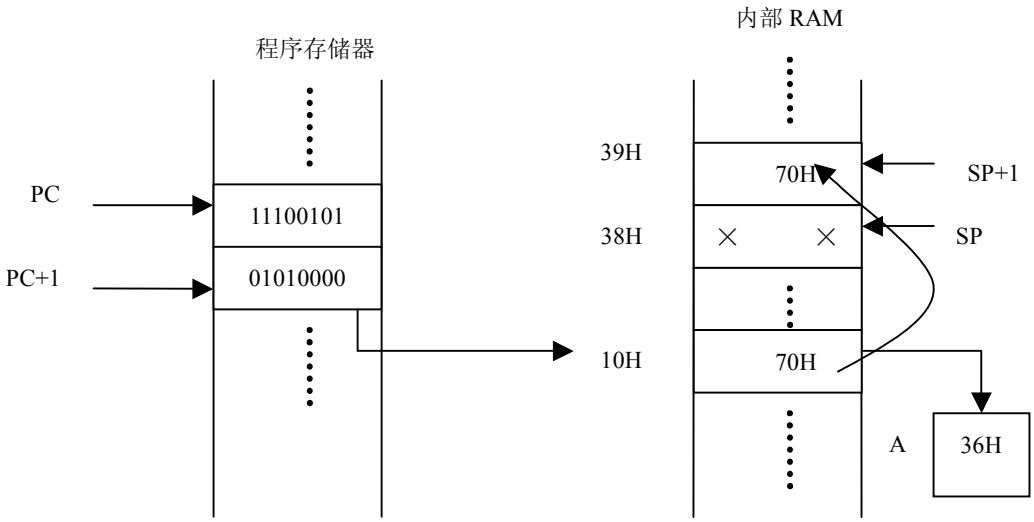


图 2-10　指令入栈操作示意图

例 2-6　已知（SP）=40H，（40H）=68H，（A）=20H

执行指令：POP　ACC

指令执行后；（SP）=3FH，（40H）=68H，（A）=68H
上述程序的指令出栈操作示意图如图 2-11 所示。

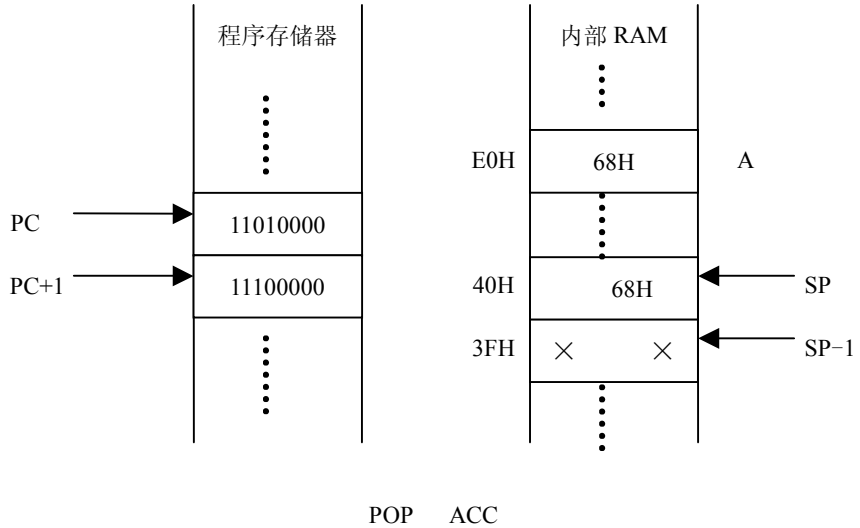


图 2-11　指令出栈操作示意图

2.4.3 数据交换指令（5 条）

数据交换指令有字节交换指令和半字节交换指令两种。字节交换指令可以将累加器 A 的内容与内部 RAM 中任一个单元以字节为单位进行交换；半字节交换指令可以将累加器 A 的高半字节和低半字节进行交换，或将累加器 A 的低半字节与@Ri 寻址单元的低半字节相互交换。数据交换指令见表 2-7 所列。

表 2-7 数据交换指令

助 记 符	功能说明	机器周期
XCH A，Rn	Rn 值和 A 值全交换，A◄►Rn。	1
XCH A，direct	direct 值与 A 值全交换，A◄►direct。	1
XCH A，@Ri	@Ri 值与 A 值全交换，A◄►Ri。	1
XCHD A，@Ri	@Ri 与 A 值的低四位交换，$A_{3\sim0}$ ◄► $(Ri)_{3\sim0}$。	1
SWAP A	A 值自交换（高 4 位与低 4 位交换），$A_{7\sim4}$◄► $A_{3\sim0}$。	1

例 2-7 已知 A=56H，R0=20H，（20H）=78H，（10H）=18H，R4=8AH

单独执行指令： ① `XCH A， 10H`

②`XCH A， R4`

③ `XCH A， @R0`

指令执行后： ① A=18H，（10H）=56H；

② A =8AH，R4=56H；

③ A=78H，R0=20H，（R0）=（20H）=56H。

例 2-8 已知 A=7AH，R1=48H，（48H）=0DH

执行指令：`XCHD A，@R1`

指令执行后：A=7DH，R1=48H，(R1)=（48H）=0AH。

2.4.4 外部 RAM 数据传送指令（4 条）

外部 RAM 数据传送指令助记符为 MOVX，表 2-8 所列为外部 RAM 数据传送指令。

表 2-8 外部 RAM 数据传送指令

助 记 符	功能说明	机器周期
MOVX A，@DPTR	用 DPTR 间址的片外 RAM 指定单元内容送入 A 中，A←（DPTR）。	2
MOVX @DPTR，A	将 A 中的内容送至片外 RAM 的 DPTR 间址的单元中，（DPTR）←（A）。	2
MOVX A，@Ri	将片外 RAM 中由 Ri 间址的地址单元中内容送入 A 中，A←（Ri）。	2
MOVX @Ri，A	将 A 中的内容送至片外 RAM 中由 Ri 间址的地址单元中，（Ri）←A。	2

注意，MSC—51 系列单片机扩展的 I/O 接口的端口地址占用的是外部 RAM 的地址空间，因此对扩展的 I/O 接口而言，这 4 条指令为 I/O（输出/输出）指令。MCS—51 系列单片机只能用这种方式与连接在扩展的 I/O 接口外部设备进行数据传送。

指令中以 DPTR 为外部 RAM 的 16 位地址指针时，由 P0 端口送出低 8 位地址，由 P2 端口送出高 8 位地址，寻址范围为 64 KB；以 R0 或 R1 为外部 RAM 的低 8 位地址指针时，由 P0 端口送出 8 位地址，P2 端口的状态不变化，寻址范围为 256 个单元（即一页）。

2.4.5　程序存储器查表指令（2 条）

程序存储器查表指令的助记符为 MOVC。程序存储器为只读存储器。程序运行中所需的一些常数和表格，通常是由用户先将其写在程序存储器中，程序需从程序存储器中读出数据时，采用变址寻址方式将表格常数读入累加器 A 中。程序存储器查表指令见表 2-9 所列。

表 2-9　程序存储器查表指令

助 记 符	功能说明	机器周期
MOVC　A，@A+DPTR	将以 DPTR 为基址，A 为偏移地址的存储单元值送入 A 中，PC←PC+1，A←（A+PC）。	2
MOVC　A，@A+PC	将以 PC 为基址，A 为偏移地址的存储单元值送入 A 中，A←（A+DPTR）。	2

例 2-9　已知 A=38H

执行指令：`1000H：MOVC　A，@A+PC`

解：该指令以 PC 内容加 1 后的当前值作为基址寄存器，累加器 A 为变址寄存器（8 位无符号整数），两者内容相加得到一个 16 位地址。将该地址对应的程序存储器单元的内容送给累加器 A。所以上面指令的操作是将程序存储器 1039H 单元的内容送给累加器 A。

说明：累加器 A 为 8 位无符号整数，表格位置（查表指令后 255 字节内）和表格长度（小于 255 字节）受限制。

例 2-10　已知 A=50H，DPTR=5000H

执行指令：　`MOVC　A，　@A+DPTR`

解：该指令以 DPTR 为基址寄存器，累加器 A 为变址寄存器，因此该指令执行的操作是将程序存储器中 5050H 单元的内容送入 A 中。

温馨提示：

以上介绍的数据传送类指令是用得最多、最频繁的一类指令，读者应熟记 8 种助记符的功能以及能采用的操作数寻址方式。

2.5　如何使用控制转移类指令

控制转移类指令共有 22 条。分为无条件转移指令（AJMP，LJMP，SJMP，JMP），条件转移指令（JZ，JNZ，JC，JNC，JB，JNB，JBC，CJNE，DJNZ），调用和返回指令（ACALL，LCALL，RET，RETI），空操作指令（NOP）。

在控制转移类指令中，除“CJNE”指令对 Cy 有影响外，其余指令都不影响标志。

控制与转移类指令可改变程序计数器 PC 的值，从而使程序跳到指定的目的地址开始执行。

2.5.1 无条件转移指令（4 条）

无条件转移指令见表 2-10 所列。程序执行无条件转移指令时，程序无条件地转移到目的地址。

表 2-10 无条件转移指令

助 记 符	功能说明	机器周期
LJMP addr16	长转移指令：程序转移到 addr16 指示的地址处，即 PC←addr16。	2
AJMP addr16	绝对转移指令：程序转移到 addr16 指示的地址处，即 PC←PC+2，$PC_{15\sim11}$ 不变，$PC_{10\sim0}$←$addr_{10\sim0}$ 。	2
SJMP rel	短转移指令：程序转移到 rel 指示相对地址处，PC←PC+2，PC←PC+rel。	2
JMP @A+DPTR	间接长转移指令（散转指令）：程序转移到 DPTR 为基址加 A 偏移地址处，即 PC←A+DPTR。	2

1. 长转移指令 LJMP addr16

指令的操作是将 16 位目标地址 addr16 装入 PC 中，允许转移的目标地址在 64KB 空间的任意单元，用汇编语言编写程序时，addr16 往往是一个标号。

2. 绝对转移指令 AJMP addr11

指令的操作是将 11 位的目标地址 addr11 装入 PC 中的低 11 位。要求目标地址的高 5 位与 PC+2 后 PC 中的高 5 位相同。即转移的目标地址必须和 AJMP 指令的下一条指令首字节地址位于程序存储器的同一段 2KB 字节范围内，编写程序时，addr11 往往也是一个标号。

3. 短转移指令 SJMP rel

指令中相对偏移量 rel 为 8 位的补码，将其符号扩展为 16 位后与 PC 相加得到 16 位的目标地址。转移的范围为−128～+127 字节，编写程序时，rel 同样往往是一个标号。

MCS—51 系列单片机中没有专用的停机指令，若要动态停机（原地循环等待）可以用 SJMP 指令来实现，如下所示。

动态停机指令；

```
LP1：SJMP  LP1
```

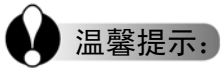
或写成：`SJMP  $`

$表示本指令首字节所在单元的地址，使用本可省略标号。

4. 间接长转移指令 JMP @A+DPTR

转移目标地址由数据指针 DPTR 和累加器 A（8 位无符号数）相加而得。指令的执行

不影响累加器 A 和数据指令 DPTR。该指令的特点是转移地址可以在程序运行中加以改变。例如 DPTR 作为基地址，根据 A 的不同值可以实现多分支转移，因此一条指令可以完成多分支转移的功能，将该功能称之为散转功能。间接长转移指令又称为散转指令。



（1） LJMP、AJMP、SJMP 三条无条件转移指令的区别如下所示。

① 转移范围不一样。LJMP 转移范围是 64KB；AJMP 转移范围是与当前 PC 值为同一 2KB；SJMP 转移范围是当前 PC：−128B～+127B。

使用 AJMP 指令和 SJMP 指令应注意转移目标地址是否在转移范围内，若超出范围，程序将出错。

② 指令字节不一样。LJMP 是 3 字节指令；AJMP 和 SJMP 都是 2 字节指令。

（2） JMP 为一字节无条件转移指令，属于变址寻址。

2.5.2　条件转移指令（13 条）

条件转移指令的操作是判断指令的条件，如果条件满足则转移，不满足则顺序执行。

条件转移指令共有 13 条，可分为判断 A 是否为零转移指令（JZ，JNZ），位条件转移指令（JC，JNC，JB，JNB，JBC），比较不等转移指令（CJNE），减“1”循环转移指令（DJNZ）。

1. 判断 A 是否为转移指令（2 条）

判断 A 是否为零指令见表 2-11 所列。

表 2-11　判断 A 是否为零指令

助 记 符	功能说明	机器周期
JZ　rel	A 值为零时，程序转移到相对地址 rel 处：若（A）≠0，则 PC←PC+2；若 A=0，则 PC←PC+2+rel。	2
JNZ　rel	A 值不为零时，程序转移到相对地址 rel 处：若 A=0，则 PC←PC+2；若 A≠0，则 PC←PC+2+rel。	2

2. 位条件转移指令（5 条）

位条件转移指令见表 2-12 所列。

表 2-12　位条件转移指令

助 记 符	功能说明	机器周期
JC　rel	进位为 0 时，程序转移至 rel：若 Cy=0，则 PC←PC+2；若 Cy=1，则 PC←PC+2+rel。	2
JNC　rel	进位为 0 时，程序转移至 rel：若 Cy=0，则 PC←PC+2+rel；若 Cy=1，则 PC←PC+2。	2
JB bit，rel	Bit 位为 1 时，程序转移至 rel 处：若 bit=0，则 PC←PC+3；若 bit=1，则 PC←PC+3+rel。	2
JNB　bit，rel	Bit 位为 0 时，程序转移至 rel 处：若 bit=0，则 PC←PC+3+rel；若 bit=1，则 PC←PC+3。	2
JBC　bit，rel	Bit 位为 1 时，程序转移至 rel 处，同时将 bit 清零：若 bit=0，则 PC←PC+3；若 bit=1，则 bit←0 后 PC←PC+3+rel。	2

3. 比较不等转移指令（4 条）

比较不等转移指令见表 2-13 所列。

表 2-13　比较不等转移指令

助 记 符	功能说明	机器周期
CJNE　A，　#data，rel	#data 与 A 内容不等时转至 rel 处，同时影响 Cy 位： 若（A）=data，则 PC←（PC）+3，Cy←0； 若（A）＞data，则 PC←（PC）+3+rel，Cy←0； 若（A）＜data，则 PC←（PC）+3+rel，Cy←1。	2
CJNE　A，　direct，rel	direct 值与 A 值不等时转至 rel 处，同时影响 Cy 位： 若（A）=（direct），则 PC←（PC）+3，Cy←0； 若（A）＞（direct），则 PC←（PC）+3+rel，Cy←0； 若（A）＜（direct），则 PC←（PC）+3+rel，Cy←1。	2
CJNE　Rn，＃data，rel	#data 与 Rn 值不等时转至 rel 处，同时影响 Cy 位： 若（Rn）=data，则 PC←（PC）+3，Cy←0； 若（Rn）＞data 则 PC←（PC）+3+rel，Cy←0; 若（Rn）＜data 则 PC←（PC）+3+rel，Cy←1。	2
CJNE　@Ri，#data，rel	data 与@Ri 值不等时转至 rel 处，同时影响 Cy 位： 若（(Ri)）=data，则 PC←（PC）+3，Cy←0； 若（(Ri)）＞data，则 PC←（PC）+3+rel，Cy←0； 若（(Ri)）＜data，则 PC←（PC）+3+rel，Cy←1。	2

比较不等转移指令的功能是比较两个数，若两者不相等则转移，相等则顺序执行。如果第二个操作数（无符号数）大于第一个操作数（无符号数），则 Cy 置 1，否则 Cy 清零。指令的执行不影响操作数。

温馨提示：

实际上 CJNE 指令执行后的 Cy 状态相当于减法指令，CJNE 指令是做了一次没有差值的减法（试减），目的操作数大于等于源操作数，Cy=0；目的操作数小于源操作数，Cy=1。

4. 减“1”循环转移指令

减“1”循环转移指令见表 2-14 所列。

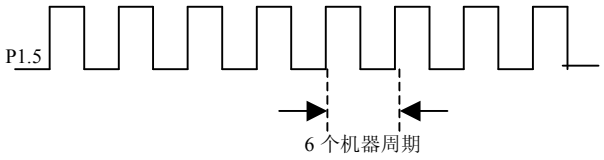
表 2-14　减“1”循环转移指令

助 记 符	功能说明	机器周期
DJNZ　Rn，rel	Rn 值先减 1，即 Rn←（Rn）−1： 若 Rn 不为零，则 PC←（PC）+2+rel，程序转移到相对地址 rel 处； 若（Rn）=0，则 PC←（PC）+2，按原顺序向下执行。	2
DJNZ　direct，rel	direct 值先减 1，即 direct←（direct）−1： 若 direct 不为零，则 PC←（PC）+3+rel，程序转移到相对地址 rel 处； 若（direct）=0，则 PC←（PC）+3，按原顺序向下执行。	2

当 direct 为端口地址 P0～P3 时，“DJNZ　direct，rel”为“读—改—写”指令。

例 2-11　从 P1.5 端口输出 8 个方波。

```
MOV  R1，#10H          ；预制方波数
LP1：CPL   P1.5        ；延时 1 个机器周期
DJNZ  R1，LP1          ；延时 2 个机器周期
```



2.5.3　调用和返回指令（4 条）

在程序设计中，通常将反复出现、具有通用性和功能相对独立的程序段设计成子程序。子程序可以有效地缩短程序长度、节约存储空间；可被其他程序共享以及便于模块化、阅读、调试和修改。子程序调用示意图如图 2-12 所示。

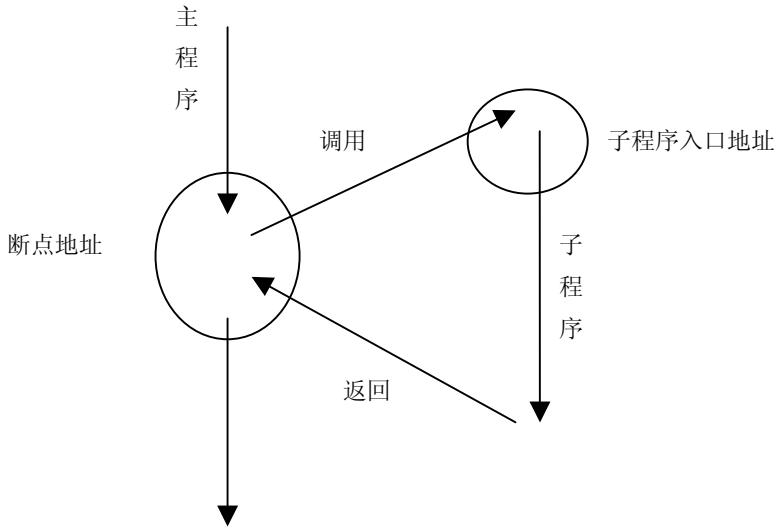


图 2-12　子程序调用示意图

子程序调用返回指令见表 2-15 所列。

长调用指令的目标地址以 16 位给出，允许子程序放在 64KB 空间的任何地方。指令的执行过程是把 PC 加上本指令代码数（三个字节）获得下一条指令的地址，并把该断点地址入栈（断点地址保护），接着将被调子程序的入口地址（16 位目标地址）装入 PC，然后从该入口地址开始执行子程序。

绝对调用指令的执行过程是 PC＋2（本指令代码为两个字节）获得下一条指令的地址，并把该断点地址（当前的 PC 值）入栈，然后将断点地址的高 5 位与 11 位目标地址（指令代码第一字节的高 3 位，以及第二字节的 8 位）连接构成 16 位的子程序入口地址，使程序

转向子程序。调用子程序的入口地址和 ACALL 指令的下一条指令的地址，其高 5 位必须相同。因此子程序的入口地址和 ACALL 指令下一条指令的第一个字节必须在同一个 2 KB 范围的程序存储器空间内。

表 2-15　子程序调用返回指令

助 记 符	功能说明	机器周期
LCALL addr16	长调用：程序调用 addr16 处的子程序： PC←PC+3；SP←SP+1；SP←$PC_{7\sim0}$ SP←SP+1；SP←$PC_{15\sim8}$；PC←addr16。	2
ACALL addr11	绝对调用：程序调用 addr11 处的子程序： PC←PC+2；SP←SP+1；SP←$PC_{7\sim0}$ SP←SP+1；SP←$PC_{15\sim8}$；PC←addr11。	2
RET	子程序返回： $PC_{15\sim8}$←(SP)；SP←SP-1；$PC_{7\sim0}$←（SP）；SP←SP-1。	2
RETI	中断返回： $PC_{15\sim8}$←(SP)；SP←SP-1；$PC_{7\sim0}$←（SP）；SP←SP-1。	2

返回指令的功能是恢复断点地址，即从堆栈中取出断点地址送给 PC，因此在子程序、中断服务程序内使用堆栈时要特别小心，一定要确认执行返回指令时，SP 指向的是断点地址，否则程序将出错。

子程序是通过 RET 指令返回主程序。

中断服务子程序是通过 RETI 指令返回。执行 RETI 指令时，清除响应中断时置位的优先级状态触发器以开放中断逻辑等。

例 2-12　已知标号 LP1 的地址为 0300H，子程序 TIME1 的入口地址为 0100H，SP=70H

执行调用指令：　`LP1：ACALL　TIME1`

指令执行后：SP =72H，(71H) =02H，(72H) =03H ，PC=0100H。

例 2-13　已知标号 LOOP 的地址为 268EH，子程序 TIME2 的入口地址为 0206H，SP=6AH

执行调用指令：　`LOOP：LCALL TIME2`

指令执行后：　SP=6CH，(6BH) =91H，(6CH) =26H，PC=0206H。

例 2-14　已知 SP =78H，(78H) =46H，(77H) =8BH；

执行指令：　`RET`

指令执行后：SP =76H，PC=468BH。

即 CPU 从 468BH 处开始执行程序，子程序必须通过 RET 指令返回主程序（通常情况下，子程序都以 RET 指令结束，但一个子程序也可以有多条 RET 指令）。

2.5.4　空操作指令

空操作指令见表 2-16 所列。空操作指令不进行任何操作。执行空操作指令时，PC＋1 指向下一条指令，占用 CPU 一个机器周期时间。NOP 指令常用于等待、延时等。

表 2-16　空操作指令

助 记 符	功能说明	机器周期
NOP	空操作：　PC←（PC）+1。	1

温馨提示：

NOP 为单机周指令，在时间上占用一个机器周期，因而在延时或者等待程序中可以起到时间微调的作用。

2.6　如何计算指令执行时间

例 2-15　编写延时 10 ms 子程序，fosc=12 MHz。

解：fosc=12 MHz，一个机器周期为 1 μs。

```
DY10ms: MOV  R6，#20；置外循环次数（MOV Rn 为 1 个机器周期指令）
DLP1:   MOV  R7，#250    ；置内循环次数（DJNZ 为 2 个机器周期指令）
DLP2:   DJNZ  R7，DLP2   ；2 个机周×250=500 机周
        DJNZ  R6，DLP1       ；500 机周×20= 10000 机周
        RET                  ；RET 指令为 2 个机器周期
```

说明：　{[(2 个机周×250)+1+2]×20+1+2}×1 μs/机器周期=10063 μs≈10 ms

例 2-16　写延时 1 s 子程序，fosc=12 MHz。

解：fosc=12 MHz，一个机器周期为 1 μs。

```
DY1S:   MOV R5, #10      ；设置外循环次数
DYS0:   MOV R6, #200     ；设置中循环次数
DYS1:   MOV R7, #250     ；设置内循环次数
DYS2:   DJNZ R7, DYS2    ；2 个机器周期×250×1 μs/机器周期=0.5 ms
        DJNZ R6, DYS1    ；0.5 ms×200=0.1s
        DJNZ R5, DYS0    ；0.1 s×10=1 s
        RET
```

温馨提示：

延时 1 s 子程序实际延时={[(2 × 250+2+1) × 200+2+1] × 10+2+1} × 1 μs

=[(100600+2+1) × 10+2+1] × 1 μs=1006033 μs

2.7 如何设计发光二极管与单片机接口电路

图 2-13 所示为发光二极管与单片机接口电路，如果要让接在 P1.0 端口的 LED1 亮起来，那么只要把 P1.0 端口的电平变为低电平就可以了；相反，如果要让接在 P1.0 端口的 LED1 熄灭，就要把 P1.0 端口的电平变为高电平。同理，接在 P1.1～P1.7 端口的其他 7 个发光二极管的点亮和熄灭的方法与 LED1 相同。因此要实现流水灯功能，我们只要将发光二极管 LED1～LED8 依次点亮或熄灭，8 只 LED 灯便会一亮一暗的做流水灯了。

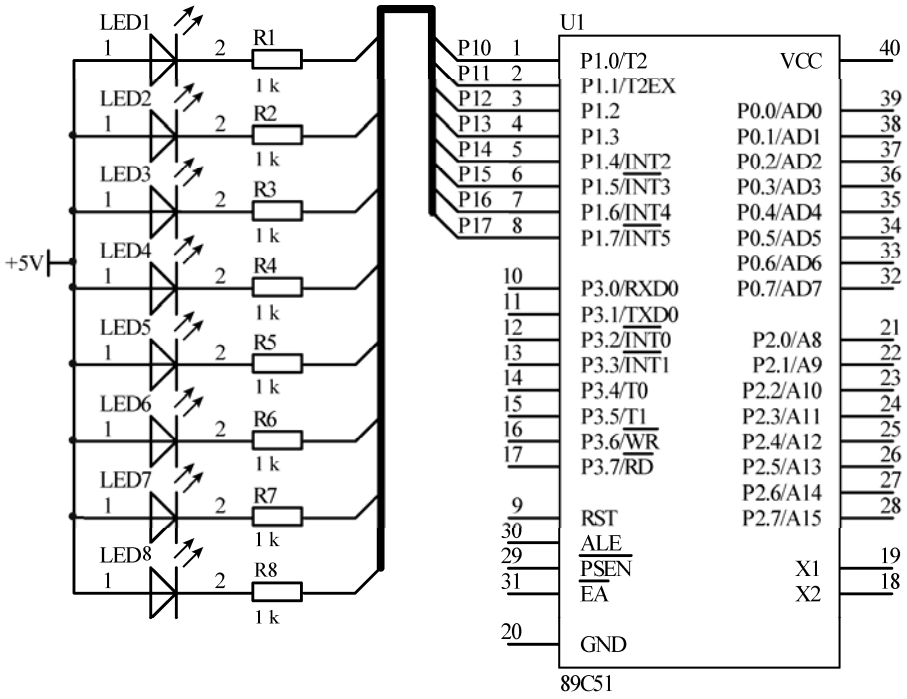


图 2-13　发光二极管与单片机接口电路

温馨提示：

由于人眼的视觉暂留效应以及单片机执行每条指令的时间很短，我们在控制二极管亮灭的时候应该延时一段时间，否则我们就看不到“流水”效果了。由于发光二极管的阳极接 VCC（假设为+5 V）的高电平，则阴极要为低电平，发光二极管才能导通发光，所以把 P1.0 端口（P1.1～P1.7 端口同理）设置为低电平就可以了。

2.8　如何设计广告灯程序

2.8.1　任务分析

单片机的应用系统由硬件和软件组成，上述硬件原理图搭建完成上电之后，还不能看到发光二极管循环点亮的现象，还需要编写程序控制单片机管脚电平的高低变化，来实现发光二极管的一亮一灭。软件编程是单片机应用系统中的一个重要的组成部分，是单片机学习的重点和难点。下面我们以最简单的控制功能即实现 8 个 LED 发光二极管的循环点亮，来介绍实现广告灯控制的几种软件编程方法。

1. 如何定时

根据以前所学知识，可以采用软件延时和定时器/计数器两种方式进行定时。软件延时方法定时，程序简单，但占用单片机运行时间，效率低；定时器/计数器定时，程序复杂，但不占用单片机运行时间，效率高。为简化程序，本任务采用软件延时方法。

2. 如何实现花样流水灯

（1） 循环移位法

利用循环移位指令，采用循环程序结构进行编程。我们在程序一开始就给 P1 端口送一个数，这个数本身让 P1.0 先处于低位，其他位为高位，然后延时一段时间，再让这个数据向高位移动，然后再输出至 P1 端口，这样就实现“流水”效果。由于 8051 型单片机的指令中只有对累加器 A 中数据左移或右移的指令，因此实际编程中我们应把需移动的数据先放到 A 中，让其移动，然后将 ACC 移动后的数据再转送到 P1 端口，这样同样可以实现“流水”效果。

（2） 查表显示法

循环移位法是比较简单的程序，“流水”花样只能实现单一的“从左到右”方式。运用查表法所编写的程序，能够实现任意方式的“流水”，只要更改其数据表的数据就可以随意添加或改变“流水”方式，真正实现随心所欲的广告灯效果。我们首先把要显示流水花样的数据建在一个以 TAB 为标号的数据表中，然后通过查表指令“MOVC　A，@A+DPTR”把数据取到累加器 A 中，然后再送到 P1 端口进行显示。具体源程序流程图如下一节所示，TAB 标号处的数据表可以根据实现效果的要求任意修改。

2.8.2　程序流程图设计

循环移位法流程图如图 2-14 所示。

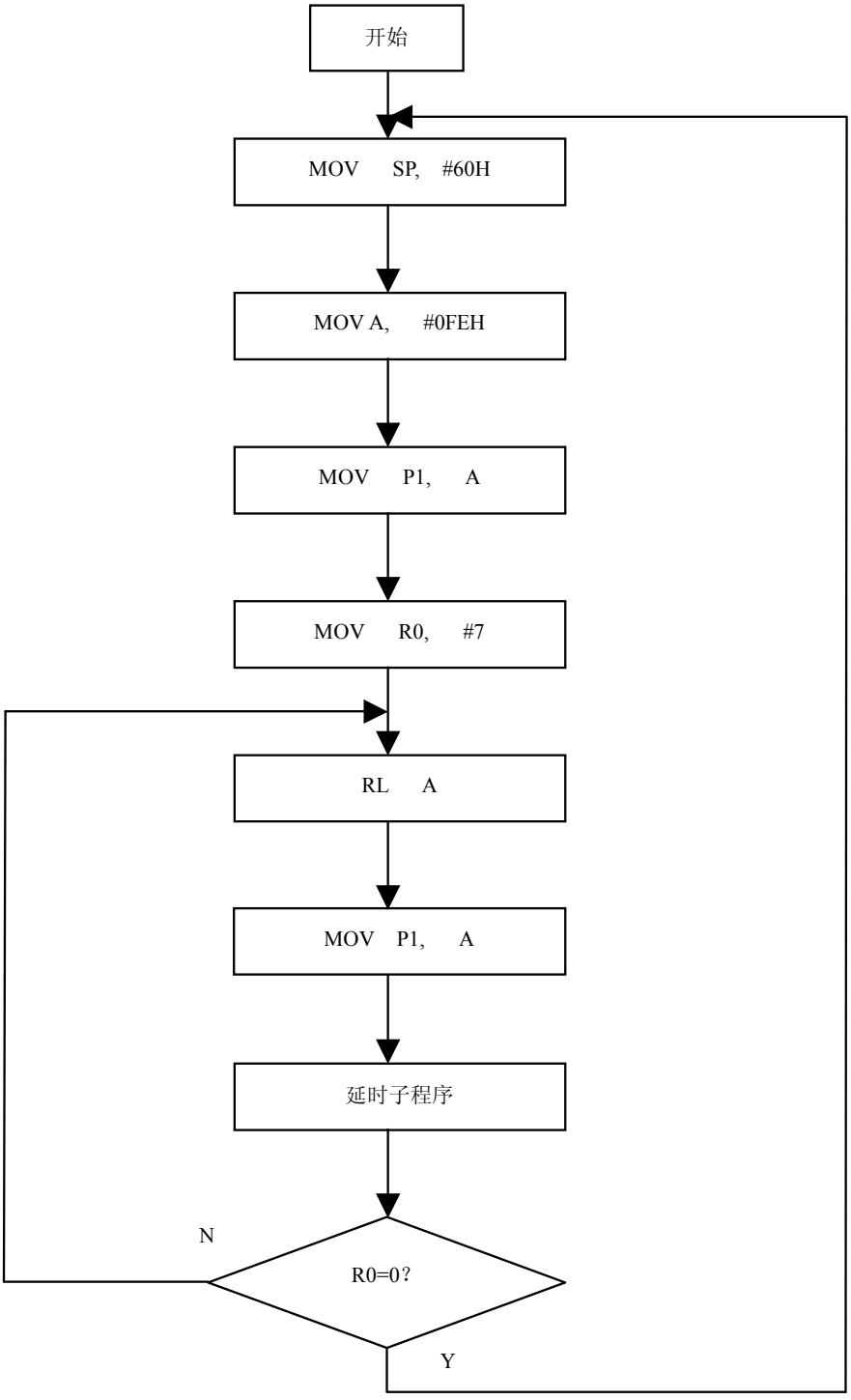


图 2-14　循环移位法流程图

查表显示法流程图如图 2-15 所示。

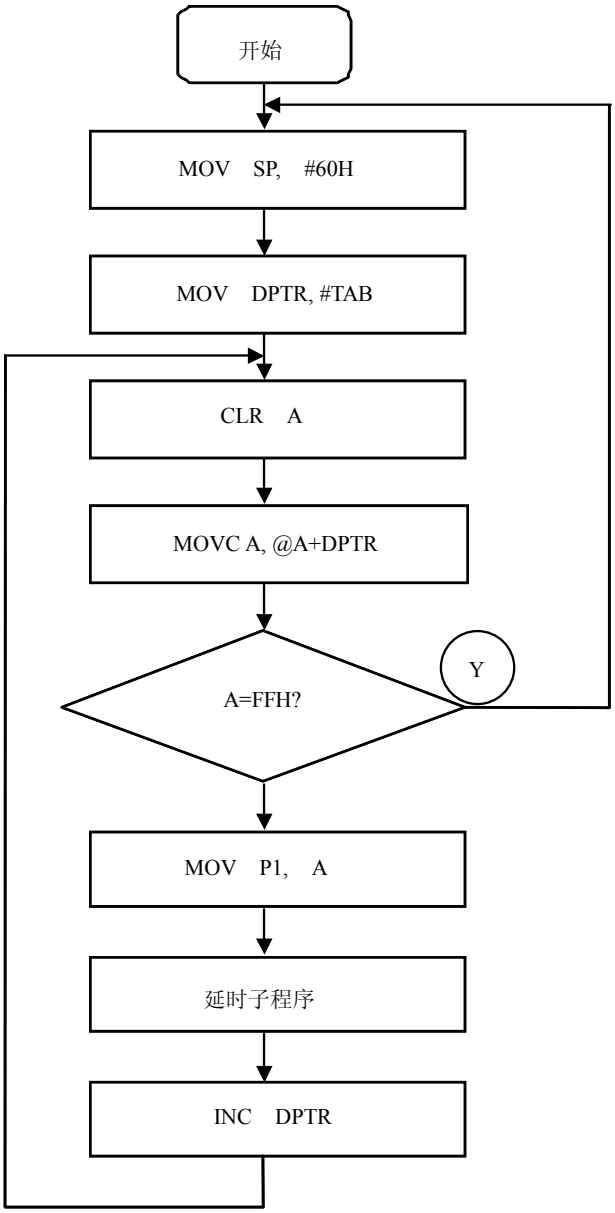


图 2-15　查表显示法流程图

2.8.3　程序清单

循环移位法程序如下所示。

```
ORG 0000H          ；单片机上电后从 0000H 地址执行
AJMP START         ；跳转到主程序存放地址处
ORG 0030H          ；设置主程序开始地址
```

```
START：
        MOV  SP，#60H          ；设置堆栈起始地址为 60H
        MOV  A，#0FEH          ；ACC 中先装入 LED1 亮的数据（二进制数 11111110）
        MOV  P1，A             ；将 ACC 的数据送 P1 口
        MOV  R0，#7            ；将数据再移动 7 次就完成一个 8 位流水过程
LOOP：
        RL A                   ；将 ACC 中的数据左移一位
        MOV P1，A              ；把 ACC 移动过的数据送 P1 端口显示
        ACALL DELAY            ；调用延时子程序
        DJNZ R0，LOOP          ；没有移动够 7 次继续移动
        AJMP START             ；移动完 7 次后跳到开始重来，以达到循环流动效果
DELAY：                        ；延时子程序
        MOV R0，#255           ；延时一段时间
D1：    MOV R1，#255
        DJNZ R1，$
        DJNZ R0，D1
        RET                    ；子程序返回
        END                    ；程序结束
```

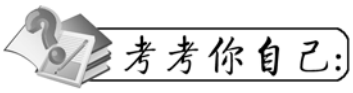
查表显示法程序如下所示。

```
        ORG 0000H              ；单片机上电后从 0000H 地址执行
        AJMP START             ；跳转到主程序存放地址处
        ORG 0030H              ；设置主程序开始地址
START：
        MOV SP，#60H           ；设置堆栈起始地址为 60H
        MOV DPTR，# TAB        ；流水花样表首地址送 DPTR
LOOP：  CLR A                  ；累加器清零
        MOVC A，@A+DPTR        ；取数据表中的值
        CJNE A，#0FFH，SHOW    ；检查流水结束标志
        AJMP START             ；所有花样流完，则从头开始重复流
SHOW：  MOV P1，A              ；将数据送到 P1 端口
        ACALL  DELAY           ；调用延时子程序
        INC  DPTR              ；取数据表指针指向下一数据
        AJMP  LOOP             ；继续查表取数据
DELAY：                        ；延时子程序
        MOV  R0，#255          ；延时一段时间
D1：    MOV R1，#255
        DJNZ  R1，$
```

```
        DJNZ  R0，D1
        RET                 ；子程序返回
TAB：                       ；下面是流水花样数据表，用户可据要求任意编写
        DB 11111110B        ；二进制数表示的流水花样数据，从低到高左移
        DB 11111101B
        DB 11111011B
        DB 11110111B
        DB 11101111B
        DB 11011111B
        DB 10111111B
        DB 01111111B
        DB 01111111B                ；二进制数表示的流水花样数据，从高到低右移
        DB 10111111B
        DB 11011111B
        DB 11101111B
        DB 11110111B
        DB 11111011B
        DB 11111101B
        DB 11111110B
        DB 0FEH，0FDH，0FBH，0F7H       ；十六进制数表示的流水花样数据
        DB 0EFH，0DFH，0BFH，7FH
        DB 7FH，0BFH，0DFH，0EFH
        DB 0F7H，0FBH，0FDH，0FEH
        ……
        DB 0FFH                     ；流水花样结束标志 0FFH
        END                         ；程序结束
```

温馨提示：

当上述程序之一编写好以后，还需要使用编译软件对其进行编译，得到单片机所能识别的二进制代码，然后再用编程器将二进制代码烧写到 AT89C51 型号的单片机中，最后连接好电路通电，我们就看到 LED1～LED8 的“流水”效果了。本章所给程序实现的功能比较简单，旨在抛砖引玉，读者可以自己在此基础上进行扩展，比如键盘控制流水方式、控制 LED 显示数字或图案等等。



1. 如何正确理解堆栈操作的“入栈”和“出栈”？
2. LJMP、AJMP、SJMP 三条无条件转移指令的区别是什么？
3. 编程：要求延时 1 ms 子程序，*f*osc=6 MHz。
4. 除了循环移位法和查表显示法，还有其他的方法可以实现广告灯的控制吗？
5. 试比较循环移位法和查表显示法的优缺点。

项目三　键盘控制显示的设计

——键盘接口技术

教学目的

掌握：键盘应用电路的设计方法。
理解：算术运算指令、逻辑运算指令和伪指令的应用；键盘驱动程序设计方法。
了解：复杂键盘的电路结构。

愿你知多点：

从消费电子、家用电器到测控仪表，都设置有键盘供操作者输入指令或调取信息，例如空调、电视的遥控器以及热水器、电饭煲的操作面板等，那么这些键盘是如何制作的呢？在这一章中，我们将通过完成键盘控制显示的设计任务来学习制作键盘的方法及相关知识。图 3-1 所示为键盘应用电路实例。

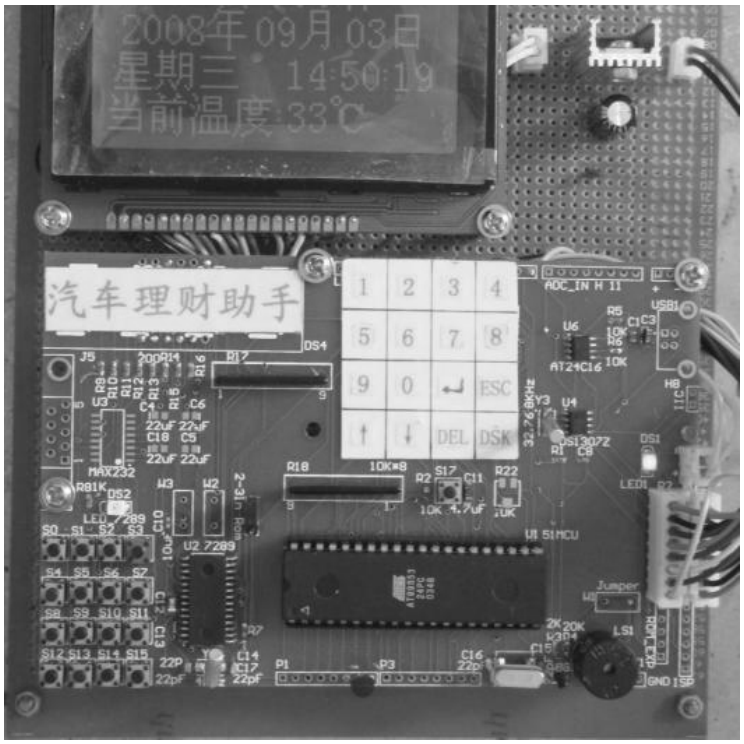


图 3-1　键盘应用电路实例

3.1　能力培养

本章通过完成键盘控制显示的设计这一任务，可以培养以下能力。

1. 能正确连接键盘电路。
2. 能正确编写检测出有无键击动作和识别按键的键值程序。
3. 能制作非编码式键盘。

3.2　任务分析

要完成此项任务，需要掌握以下 3 方面知识。

1. 如何将键击动作转换为位数字量信息。
2. 如何设计键盘与单片机接口电路。
3. 如何设计键盘驱动程序实现按键的键值计算。

3.3　如何将键击动作转换为位数字量信息

3.3.1　如何使用键盘

键盘是由若干个常开型按钮组成的开关电路，是单片机测控系统常用的输入模块，能实现人机对话，为操作者对单片机测控系统的状态干预、数据输入及控制指令输入提供途径。

键盘根据其结构不同分为编码式键盘和非编码式键盘两种，编码式键盘是由其内部硬件逻辑电路自动产生被按按键的编码值，如个人计算机的键盘，这种键盘使用方便，但价格较贵，在单片机测控系统中较少使用，因而本章主要讨论非编码式键盘。

根据非编码式键盘与单片机的接法不同，可分为行列式键盘和独立式键盘，两者的工作原理相同，都是将键击动作导致的按钮闭合状态和单片机管脚电平转换为 I/O 端口的位数字量，识别方法需要由单片机开发者编写键盘驱动程序来识别按键的键击闭合状态，辨别被按下的按键并为其分配键值。其中独立式键盘的结构和驱动程序较简单，但与行列式键盘比较占用 I/O 端口资源因而按键数量有限。上述描述中，单片机测控应用的键盘电路关键要解决两个问题，即键击抖动、按键识别（包括键击动作识别和键值表示）。

温馨提示：

编码式键盘通常由专用的集成电路扫描驱动，如 8279 型集成电路可驱动由 64 个按键以 8×8 方式连接的键盘，该芯片内部有振荡电路外加晶体为行线、列线的扫描提供的时钟频率，在此时钟作用下识别键值并存入芯片内部缓冲器，通过并行接口向单片机输送键码。

3.3.2　如何消除键盘抖动与转换位数字量

常用于键盘的按钮有 4×4 轻触开关，根据适用场合不同有贴片式、直插式等，各种按键的实物图如图 3-2 所示。

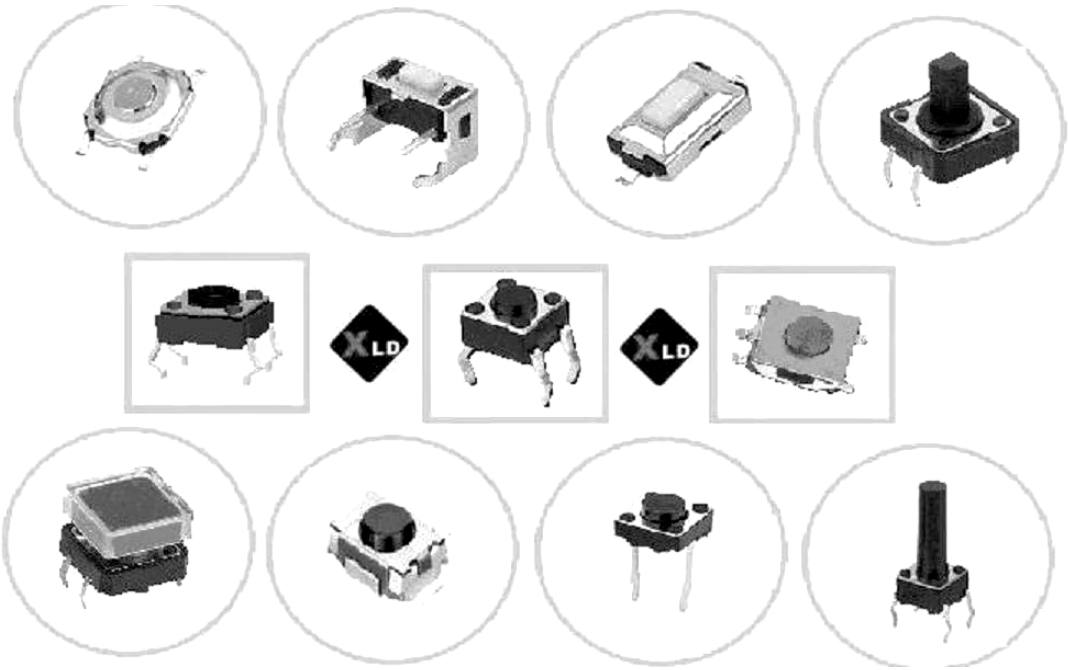


图 3-2　各种按键的实物图

这些按键是机械弹性触点式，内部安装有弹片，当施加外力克服弹片阻力时，触点闭合接通按键的两个引脚，当撤除外力弹片复原时，触点断开而引脚分离，在闭合、断开的过程中，由于机械弹性振动造成按键抖动，从而带来电气振荡。图 3-3 所示为按键回路及抖动引起的电压振荡波形图，其中图 3-3（a）所示为由按键及电阻器串联构成的一个回路，回路的中点 P1.0 在按键按下与释放过程中由于键抖动引起的电压波形振荡如图 3-3（b）所示。

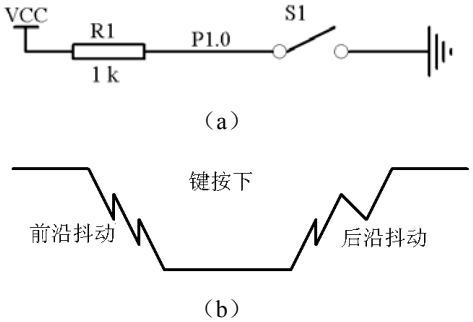


图 3-3　按键回路及抖动引起的电压振荡波形图

从图 3-3 中可以看出，抖动时间的长短由按键的机械特性决定，一般经过 5～10 ms 的振荡，电压稳定下来之后才能作为表示位数字逻辑的电平信号被检测使用，因此需要对按键采取消抖措施，通常有软件和硬件两种消抖方法。软件消抖方法是在检测按键电平后延时 10 ms 再进行一次检测，若前后一致则认为处于键稳定区。图 3-3（a）所示的电路若检测到 P1.0 点的电平为低电平则可认为 S1 被按下，此时可用位数字 0 表示键击动作。软件消抖方法简单灵活较常采用，而硬件方法只适用于按键数量较少的场合，需采用各种触发器来消抖，因此较少采用。

温馨提示：

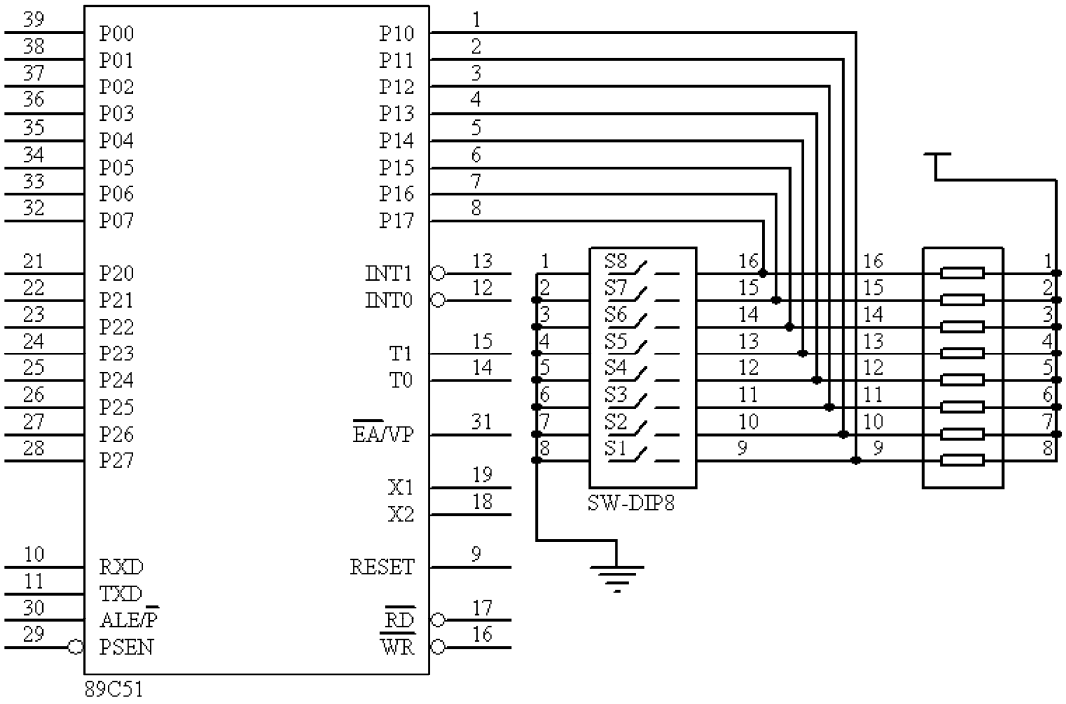
轻触开关为了增强机械强度设置有 4 个脚，其中两个是常开触点，用于连接电路，可使用万用表的 1 k 欧姆挡检测。当有触动时电气引脚导通电阻值为零，而另外两个不因触动而改变电阻器，只用于安装固定，设计电路时其焊盘不需要连接导线。

3.3.3 识别按键与计算键值

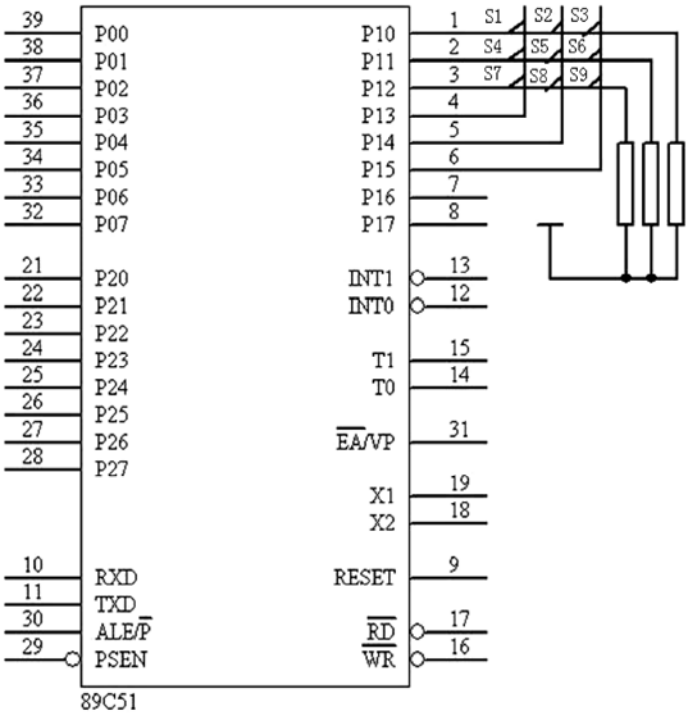
如前所述，键击动作发生时将引起电平的变化，那么如何通过电平来识别按键呢？主要通过位数字量来识别键击和表示键值。

非编码式键盘分为独立式键盘和行列式键盘，本节将以这两种类型的键盘为例，分别说明如何检测键击动作，以及如何识别按键，它们的键盘电路如图 3-4 所示，其中图 3-4（a）所示为独立式键盘，它有 8 个按键，由单片机 89C51 的 P1 端口驱动；图 3-4（b）所示为行列式键盘，属 3×3 型，即由 3 行、3 列共计 9 个按键组成，并由单片机 89C51 的 P1 端口的 PIN0～PIN2 进行列驱动，PIN3～PIN5 进行列驱动。

在独立式键盘电路中，单片机通过 I/O 端口的 P10～P12 分别连接到 3 个按键—电阻器回路的中点，当 8 个按键中的任意一个被按下时，P10～P12 对应的 PIN 引脚位电平即刻转换为低电平，若单片机读取 P1 端口的状态可获取位数字 0，否则读取到位数字 1，因此只要读取 P1 端口的电平逻辑状态基础上，利用位逻辑值 1 分别对 P1 端口上锁存器作按位逻与运算，根据与的结果是否为逻辑假（0）即可判断出各个脚位连接的按键是否被按下。独立式键盘识别键值具体流程如图 3-5 所示。



（a）独立式键盘的电路



（b）行列式键盘的电路

图 3-4　非编码式键盘的两种键盘电路

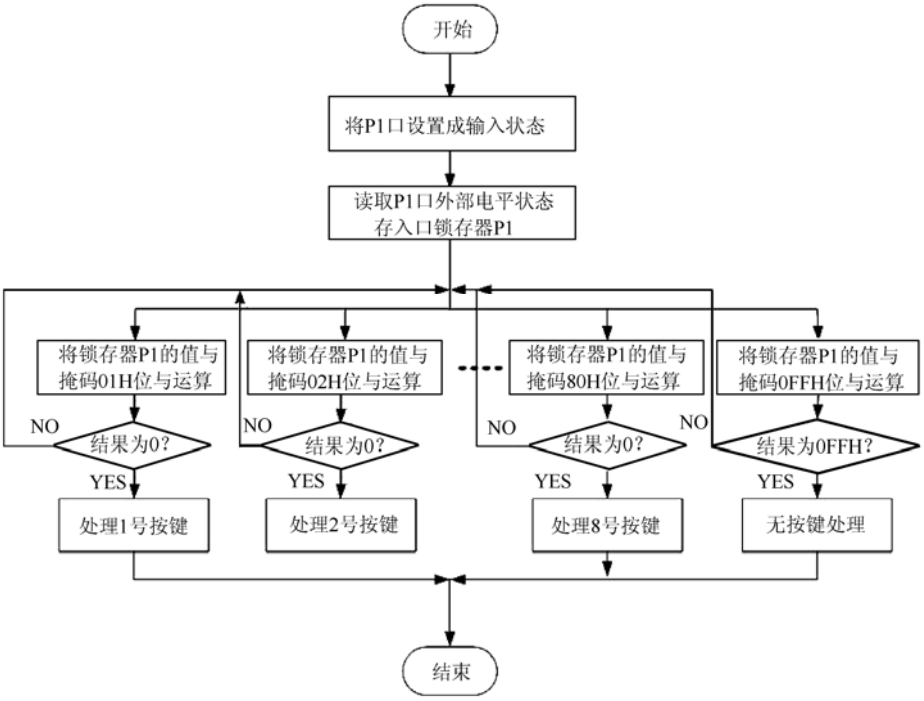


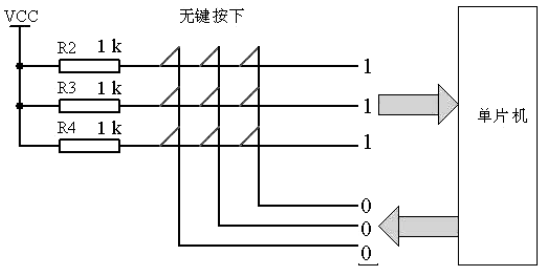
图 3-5　独立式键盘识别键值具体流程

在行列式键盘电路中，由 I/O 端口的 P10～P12 作为行线输入，而 P13～P15 作为列线，将 9 个按键组织成 3×3 矩阵，电路的结构形式与独立式键盘相比，行线 I/O 端口与按键—电阻器回路的连接位置未变，只是每个 I/O 端口连接了同一行中的 3 个按键，3 条列线相当于地从而形成一个“浮动地”，可根据需要通过单片机的逻辑运算指令将这些点分别设置为低电平（地点位）或高电平。因此，可通过列扫描的方式识别按键。图 3-6 所示为行列式键盘按键电平变化及列扫描法识别键值。

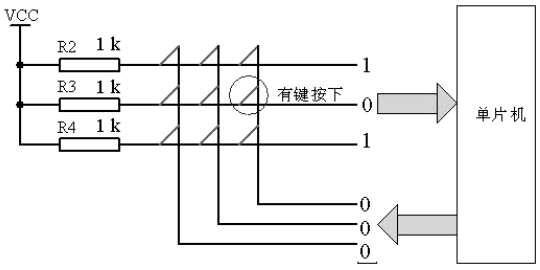
在图 3-6（a）中，当没有按键被按下时，单片机 3 条列线全部输出为 0，则将从行线读入的位数字全为 1。

在图 3-6（b）中，当圆圈中的键被按下时，列线全部输出为 0，被按下按键所在行线读入位数字为 0，其他行线仍然为 1，这样可通过检测行线状态判断是否有按键被按下，并且可知被按下按键所在行号为 i=2。

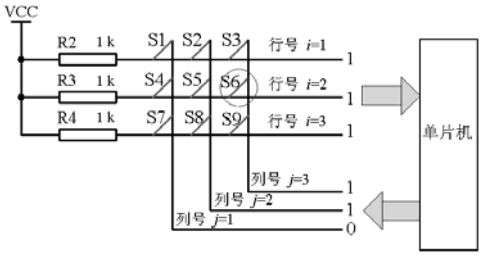
当判断有键按下后，如何让单片机进一步判断该行的 3 个按键中哪一个被按下呢？方法是三条列线中只有一条输出 0，其他列线输出 1，然后检测行线（尤其是 i=2 的行）的位数字是否出现 0，若没有则依次使下一列线输出 0，重复行线检测，直到有键按下的行（i=2）位数字为 0 为止，从而实现确定按键的位置，如图 3-6（c）～图 3-6（e）所示。通过列扫描，$i=2$，$j=2$，那么键值可由公式 $k=(i-1)\times 3+j$ 求得。行列式键盘的键值识别流程如图 3-7 所示。



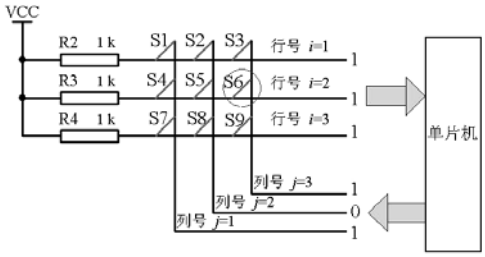
（a）



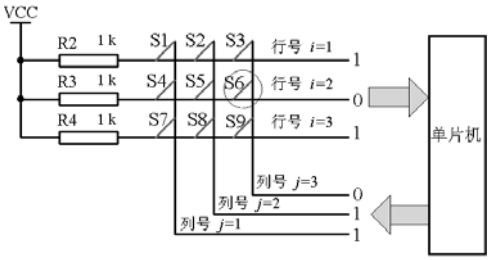
（b）行列式键盘有、无键按下时电平变化图



（c）扫描第 1 列



（d）扫描第 2 列



（e）扫描第 3 列

图 3-6　行列式键盘按键电平变化及列扫描法识别键值

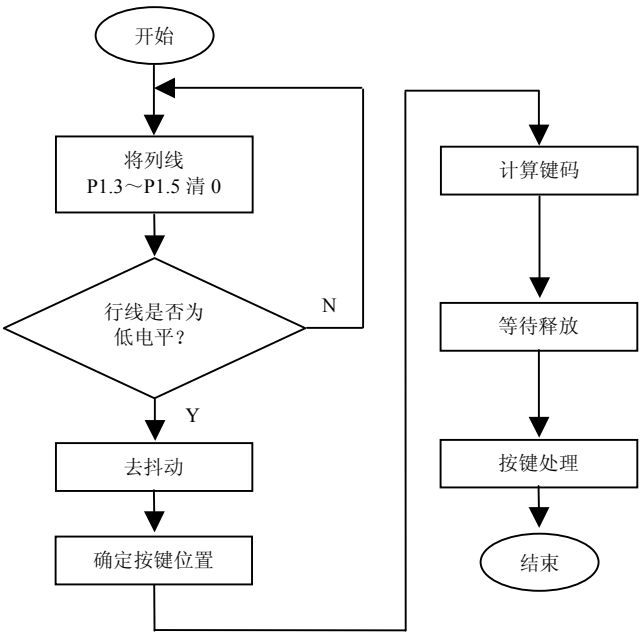


图 3-7　行列式键盘键值识别流程

温馨提示：

键值识别步骤较多，操作者是否需要长久按着按键不放呢？答案是否定的。人的反应时间在数十毫秒，而对于 8 位单片机通常采用频率为 12 MHz 的晶体振荡器，执行指令速度在 1 MIPS，即使采用列扫描方式识别按键，包括消抖延时、键值计算在内也比人的反应速度快，所以大可不必担心按键的反应速度。

然而，对于列扫描方式识别按键需要 CPU 持续不断地执行扫描任务，这将极大的消耗 CPU 的资源，如果 CPU 执行的任务较多则可能出现按下按键时并无响应，造成灵敏度降低，这些问题可通过硬件接口电路和程序优化设计改进提高。

3.4　如何设计键盘与单片机接口电路

3.4.1　独立式键盘与单片机接口电路—键盘控制显示任务

图 3-8 所示为独立式键盘与单片机接口电路应用实例，图中按键 S1、S2、S3、S4 键分别与电阻器串联接到电源和低电位点形成回路，其中电阻器与按键连接点接至单片机 89C51 的 P14、P15、P16、P17 端口，通过按键的闭合与断开状态可改变 P14、P15、P16、P17 端口的电平状态，从而改变相应接口的位值。例如，当 S1 键被按下闭合时回路导通，使得低电位点引至 P14 端口，该位值由 1 变为 0。另外，本电路中发光二极管 LED1～LED4 通过串接电阻器一端接至电源，另一端接至 I/O 端口的 P10～P13，如此可由 I/O 端口输出

位数字 1 或 0 控制对应的 LED 灯熄灭或点亮。对于本应用任务实例，可通过编程实现某个按键被按下时对应的 LED 灯点亮进行指示。

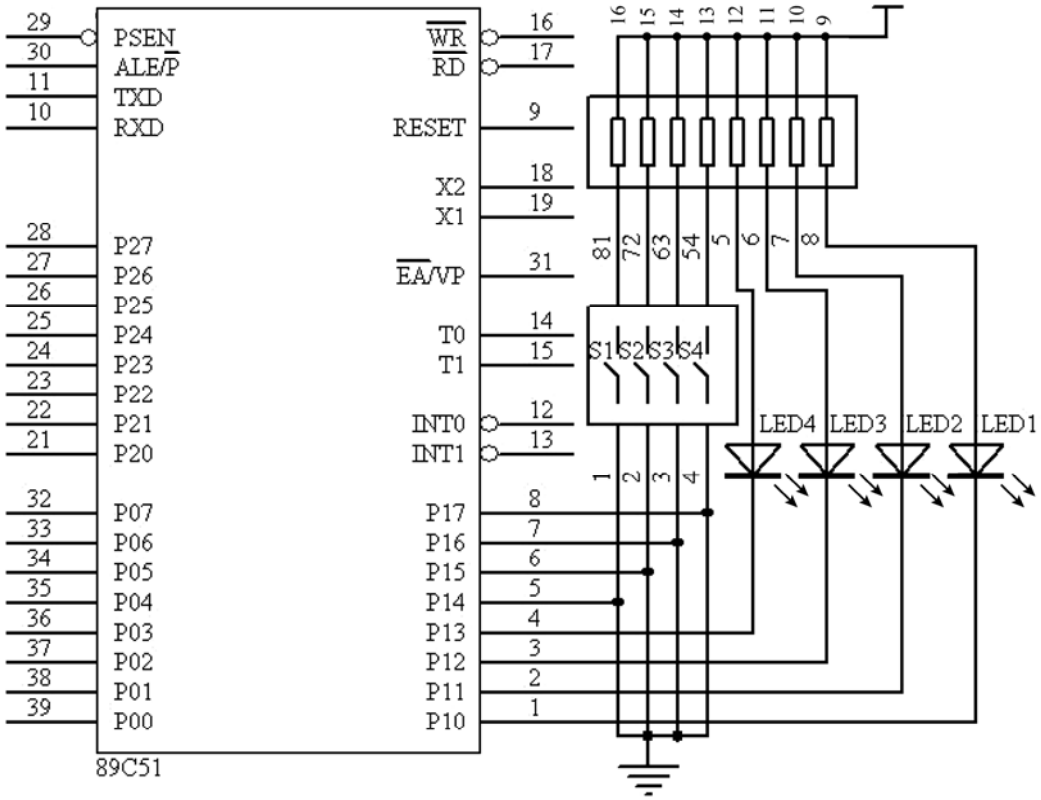


图 3-8　独立式键盘与单片机接口电路应用实例

3.4.2　行列式键盘与单片机接口电路

行列式键盘较独立式键盘对 I/O 端口的利用率高，按键数量成平方数增长，因而在复杂的单片机测控系统中较常采用，与单片机的接口电路最简单的如图 3-4（b）所示，除此之外还可灵活使用 I/O 端口构成行列式键盘，进一步节省 I/O 资源以及改进按键的识别灵敏度，降低单片机 CPU 执行按键扫描的成本。

图 3-9 所示为串行端口驱动的行列式键盘接口电路，通过使单片机串行端口工作在同步移位输出方式驱动 164 串并转换实现列扫描，这种键盘更加节省资源，当行线为 n 时可容纳 $n\times8$ 个按键。

图 3-10 所示为外部中断驱动的高灵敏度、高效率的行列式键盘接口电路，该电路在列扫描的基础上将按键的电平变化转换为外部中断信号，如此 CPU 不必保持扫描状态来识别按键，只要进入中断再启动列扫描即可，这样做一方面极大地降低 CPU 的负荷，另一方面极大地提高按键的灵敏度。

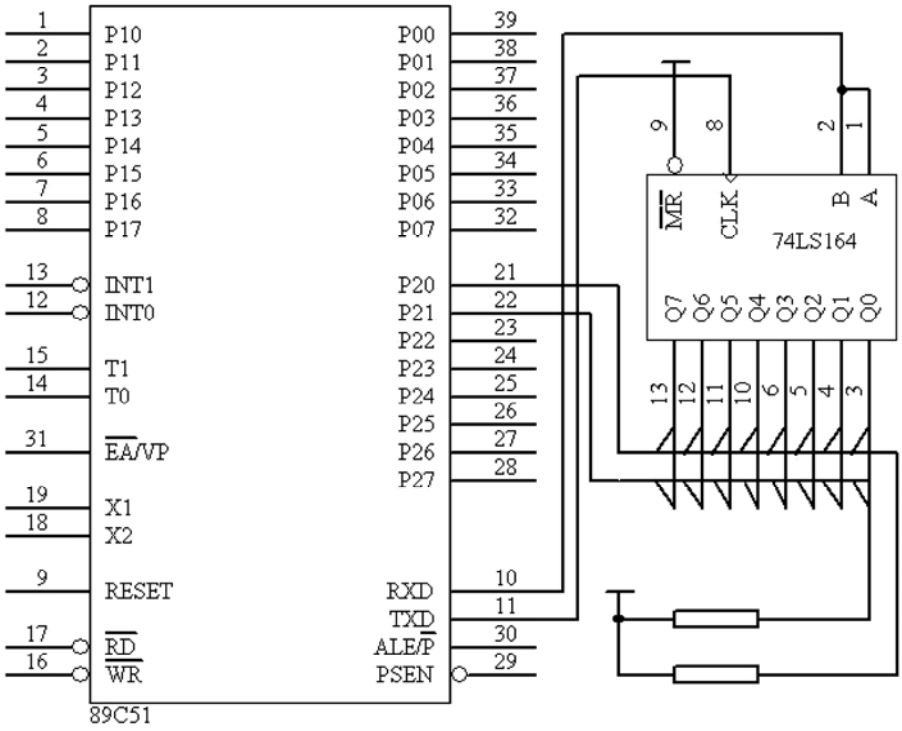


图 3-9　串行端口驱动的行列式键盘接口电路

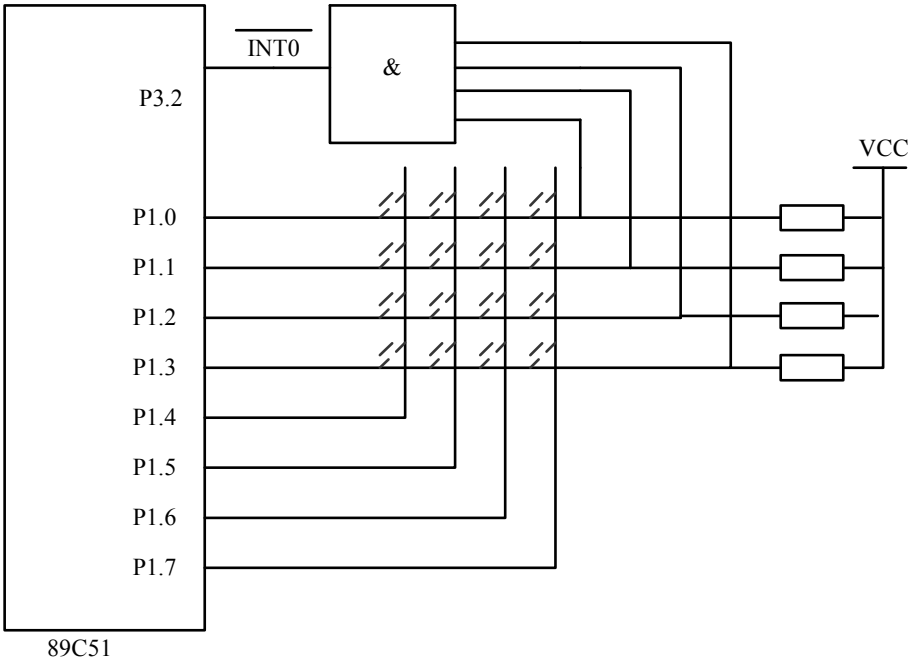


图 3-10　外部中断驱动的高灵敏度、高效率的行列式键盘接口电路

3.5　如何使用算术运算指令

在按键识别时，通过检测位数字识别按键所在的行和列如何求取键值呢？这需要使用算术和逻辑运算指令进行处理。51 系列单片机具有丰富的算术运算指令，包括加、减、乘、除、加 1 和减 1 等各类算术运算，共有 24 条。

3.5.1　加法指令

1.　不带进位的加法指令（ADD）

不带进位的加法指令见表 3-1 所列。

表 3-1　不带进位的加法指令

助 记 符	功能说明	机器周期
ADD A，Rn	A 中的内容与 Rn 中的内容相加，结果存于 A 中：A←A+Rn	1
ADD A，direct	A 中的内容与 direct 中的内容相加，结果存于 A 中：A←A+(direct)	1
ADD A，#data	A 中的内容与立即数相加，结果存于 A 中：A←A+data	1
ADD A,@Ri	A 中的内容与 Ri 间址的地址单元中内容相加，结果存于 A 中：A←A+(Ri)	1

例如 A=53H，（70H）=0FCH。

```
ADD A，70H
```

结果 A=4FH，Cy=1。

2.　带进位的加法指令（ADDC）

带进位的加法指令见表 3-2 所列。

表 3-2　带进位的加法指令

助 记 符	功能说明	机器周期
ADDC A，Rn	A 中的内容与 Rn 中的内容以及 Cy 相加，结果存于 A 中：A←A+Rn+Cy	1
ADDC A，direct	A 中的内容与 direct 中的内容以及 Cy 相加，结果存于 A 中：A←A+(direct) +Cy	1
ADDC A，#data	A 中的内容与立即数以及 Cy 相加，结果存于 A 中：A←A+data+Cy	1
ADDC A，@Ri	A 中的内容与 Ri 间址的地址单元中内容以及 Cy 相加，结果存于 A 中：A←A+(Ri) +Cy	1

例如被加数放在 20H、21H 单元，加数放在 30H、31H 单元，其值分别为 80H、90H、A0H、B0H，要求所得的和存放在 4000H、4001H 单元。

```
CLR C
MOV A,20H
ADD A,30H
MOV DPTR,#4000H
MOVX @DPTR,A
```

```
MOV A,21H
ADDC A,31H
MOV DPTR,#4001H
MOVX @DPTR,A
```

执行后，Cy=1，A=41H，（4000H）=20H，（4001H）=41H。

温馨提示：

带进位加法指令 ADDC 有三个运算数，Cy 的值参与相加，常用于双字节加法运算中。先对低字节数执行 ADD 指令，进位记入 Cy，然后再对高字节数作带进位的加法运算，所以在执行 ADD 指令之前需要对 Cy 清零。

3. 加 1 指令（INC）

加 1 指令见表 3-3 所列。

表 3-3 加 1 指令

助 记 符	功能说明	机器周期
INC A	A 中的内容加 1，结果存于 A 中：A←A+1	1
INC Rn	Rn 中的内容加 1，结果存于 Rn 中：Rn←Rn+ 1	1
INC direct	Direct 中的内容加 1，结果存于 Direct 中：(direct) ← (direct)+1	1
INC @Ri	Ri 间址的地址单元中内容加 1，结果存于 Ri 间址的地址单元中：(Ri)←(Ri)+1	1
INC DPTR	DPTR 中的内容加 1，结果存于 DPTR 中：DPTR←DPTR+1	2

例如
```
MOV A，#00H
INC A
```

执行后，A=01H。

3.5.2 减法指令

1. 带借位的减法指令（SUBB）

带借位的减法指令见表 3-4 所列。

表 3-4 带借位的减法指令

助 记 符	功能说明	机器周期
SUBB A，Rn	A 中的内容与 Rn 中的内容以及 Cy 相减，结果存于 A 中：A←A-Rn- Cy	1
SUBB A，direct	A 中的内容与 direct 中的内容以及 Cy 相减，结果存于 A 中：A←A-direct- Cy	1
SUBB A，,#data	A 中的内容与立即数以及 Cy 相减，结果存于 A 中：A←A-data-Cy	1
SUBB A，@Ri	A 中的内容与 Ri 间址的地址单元中内容以及 Cy 相减，结果存于 A 中：A←A-(Ri)- Cy	1

例如被减数放在 30H～32H 单元，减数放在 40H～42H 单元，其值分别为 50H、60H、70H、80H、90H 和 A0H，要求差存放入 50H～52H 单元。

```
CLR C
MOV A，30H
SUBB A，40H
MOV 50H，A
MOV A，31H
SUBB A，41H
MOV 51H，A
MOV A，32H
SUBB A，42H
MOV 52H，A
```

执行后，Cy=1，（50H）=F0H，（51H）=EFH，（52H）=EFH。


温馨提示：

带借位减法指令 SUBB 有三个运算数，Cy 的值参与其中，在执行 SUBB 指令之前应根据需要对 Cy 清零。

2. 减 1 指令（DEC）

减 1 指令见表 3-5 所列。

表 3-5　减 1 指令

助 记 符	功能说明	机器周期
DEC A	A 中的内容减 1，结果存于 A 中：A←A-1	1
DEC Rn	Rn 中的内容减 1，结果存于 Rn 中：Rn←Rn- 1	1
DEC direct	direct 中的内容减 1，结果存于 direct 中：(direct) ← (direct)-1	1
DEC @Ri	Ri 间址的地址单元中内容减 1，结果存于 Ri 间址的地址单元中：(Ri)←(Ri)-1	1

3.5.3　十进制数据调整指令

十进制数据调整指令见表 3-6 所列。

表 3-6　十进制数据调整指令

助 记 符	功能说明	机器周期
DA　A	对 A 的内容进行 BCD 码调整	1

指令对 BCD 码加法运算的结果自动修正，使结果为压缩的 BCD 码，通常在对 BCD 码执行 ADD 或 ADDC 后使用。

例如设 A=78H，R3=25H，

执行指令：
```
ADD A，R3
DA  A
```

结果为：A=03H，Cy=1。

3.5.4 乘法指令

乘法指令见表 3-7 所列。

表 3-7 乘法指令

助 记 符	功能说明	机器周期
MUL AB	A 中的内容与 B 中的内容相乘，结果低 8 位存入 A 中，高 8 位存入 B 中：BA←A×B	4

温馨提示：

A×B，结果是 16 位，高 8 位存入 B 中，低 8 位存入 A 中，若乘积大于 FFH 则将溢出标志 OV 置 1。

例如设 A=4EH，B=5DH，
执行指令：`MUL  AB`
结果为：A=56H，B=1CH，OV=1，Cy=0。

3.5.5 除法指令

除法指令见表 3-8 所列。

表 3-8 除法指令

助 记 符	功能说明	机器周期
DIV AB	A 中的内容除以 B 中的内容，商存入 A 中，余数存入 B 中：A（商），B（余数）←A÷B	4

温馨提示：

A ÷ B，商存入 A，余数存入 B，当除数为 0 时，结果不确定，则将溢出标志 OV 置 1。

3.6 如何使用逻辑运算指令

51 系列单片机拥有丰富的逻辑运算指令，可进行清除、求反、移位、与、或、异或等操作，共有 17 条指令。

1. 逻辑与指令（ANL）

逻辑与指令见表 3-9 所列。

表 3-9　逻辑与指令

助 记 符	功能说明	机器周期
ANL　A，Rn	A 中的内容与 Rn 中的内容相与，结果存于 A 中：A←A ∧Rn	1
ANL　A，direct	A 中的内容与 direct 中的内容相与，结果存于 A 中：A←A ∧（direct）	1
ANL　A，@Ri	A 中的内容与 Ri 间址的地址单元中内容相与，结果存于 A 中：A←A ∧（Ri）	1
ANL　A，#data	A 中的内容与立即数相与，结果存于 A 中：A←A ∧data	1
ANL　direct，A	direct 中的内容与 A 中的内容相与，结果存于 direct 中：（direct）←（direct）∧A	1
ANL　direct，#data	direct 中的内容与立即数相与，结果存于 direct 中：（direct）←（direct）∧data	2

例如若 A=95H=10010101B，以下两条指令：

```
ANL　A，#0FH
ANL　A，#0F0H
```

结果分别为 05H 和 90H。

2. 逻辑或指令（ORL）

逻辑或指令见表 3-10 所列。

表 3-10　逻辑或指令

助 记 符	功能说明	机器周期
ORL　A，Rn	A 中的内容与 Rn 中的内容相或，结果存于 A 中：A←A ∨Rn	1
ORL　A，direct	A 中的内容与 direct 中的内容相或，结果存于 A 中：A←A ∨（direct）	1
ORL　A，@Ri	A 中的内容与 Ri 间址的地址单元中内容相或，结果存于 A 中：A←A ∨（Ri）	1
ORL　A，#data	A 中的内容与立即数相或，结果存于 A 中：A←A ∨data	1
ORL　direct，A	direct 中的内容与 A 中的内容相或，结果存于 direct 中：（direct）←（direct）∨A	1
ORL　direct，#data	direct 中的内容与立即数相或，结果存于 direct 中：（direct）←（direct）∨data	2

例如若 A=95H=10010101B，以下两条指令：

```
ORL　A，#0FH
ORL　A，#0F0H
```

结果分别为 9FH 和 F5H。

3. 逻辑异或指令（XRL）

逻辑异或指令见表 3-11 所列。

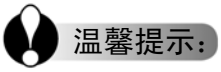
表 3-11　逻辑异或指令

助 记 符	功能说明	机器周期
XRL　A，Rn	A 中的内容与 Rn 中的内容相异或，结果存于 A 中：A←A ⊕Rn	1
XRL　A，direct	A 中的内容与 direct 中的内容相异或，结果存于 A 中：A←A ⊕（direct）	1
XRL　A，@Ri	A 中的内容与 Ri 间址的地址单元中内容相异或，结果存于 A 中：A←A⊕（Ri）	1
XRL　A，#data	A 中的内容与立即数相异或，结果存于 A 中：A←A ⊕data	1
XRL　direct，A	direct 中的内容与 A 中的内容相异或，结果存于 direct 中：（direct）←（direct）⊕A	1
XRL　direct，#data	direct 中的内容与立即数相异或，结果存于 direct 中：（direct）←（direct）⊕data	2

例如 A=95H=10010101B，

执行指令：XRL　A，#95H

结果：A=0。



和 0 相与使被修改数的相应位清零，和 1 相与使被修改数的相应位保持不变；和 1 相或使被修改数的相应位置 1，和 0 相或使被修改数的相应位保持不变；与 1 相异或使被修改数的相应位取反，和 0 相异或使被修改数的相应位保持不变，此指令可比较两数是否相等，若相等则得到结果为 0，否则两数不相等。设计合适的掩码参与逻辑运算可对变量进行灵活的位操作。

4. 累加器 A 的逻辑操作指令

累加器 A 的逻辑操作指令见表 3-12 所列。位操作指令也称布尔变量操作指令，其操作对象为内部 RAM 位寻址区以及可以进行位寻址的 SFR。位操作指令包括位传送、位置 1、位清零、位运算和位控制转移指令 4 类。

表 3-12　累加器 A 的逻辑操作指令

助 记 符	功能说明	机器周期
CPL A	将 A 内容按位逐位取反，结果存于 A 中：A← $\overline{A}$	1
CLR A	将 A 内容清零，结果存于 A 中：A←00H	1

1. 位传送指令

位传送指令见表 3-13 所列。

表 3-13　位传送指令

助 记 符	功能说明	机器周期
MOV C，bit	将 bit 传送给 PSW 中的进位标志 Cy ：Cy←bit	1
MOV bit，C	将 PSW 中的进位标志 Cy 传送给 bit： bit←Cy	2

2. 位置 1 和位清零指令

位置 1 和位清零指令见表 3-14 所列。

表 3-14　位置 1 和位清零指令

助 记 符	功能说明	机器周期
CLR C	将 PSW 中的进位标志 Cy 清零 ：Cy←0	1
CLR bit	将 bit 位清零：bit←0	1
SETB C	将 PSW 中的进位标志 Cy 置 1：Cy←1	1
SETB bit	将 bit 位置 1：bit←1	1

3. 位运算指令

位运算指令见表 3-15 所列。

表 3-15　位运算指令

助 记 符	功能说明	机器周期
ANL C，bit	Cy 与 bit 相与，结果送给 Cy：Cy←Cy∧bit	2
ANL C，/bit	Cy 与 bit 非后相与，结果送给 Cy ：Cy←Cy∧ $\overline{bit}$	2
ORL C，bit	Cy 与 bit 相或，结果送给 Cy：Cy←Cy∨ bit	2
ORL C，/bit	Cy 与 bit 非后相或，结果送给 Cy： Cy←Cy∨ $\overline{bit}$	2
CPL C	Cy 取反：Cy← $\overline{Cy}$	2
CPL bit	bit 取反：bit← $\overline{bit}$	2

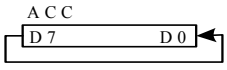
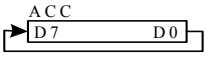
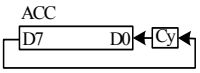
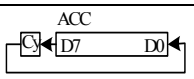
4. 位控制转移指令

位控制转移指令已在 2.5.2 节条件转移指令做了详细介绍，这里不再重复。

3.7　如何循环移位指令

循环移位指令见表 3-16 所列。

表 3-16　循环移位指令

助 记 符	功能说明	机器周期
RL　A	A 内容循环左移： 	1
RR　A	A 内容循环右移： 	1
RRC　A	A 内容与 CY 内容一起循环右移： 	1
RLC　A	A 内容与 CY 内容一起循环左移： 	1

例如 A=95H=10010101B，Cy=1，

执行指令：RRC　A

结果：A=2BH。

3.8　如何使用伪指令

伪指令是不能执行的指令，在程序汇编时起控制作用。

1. 汇编起始命令

格式：　ORG　16 位地址

ORG 常用于汇编语言源程序或数据块的开头，用来指示汇编程序开始对源程序进行

汇编。

2. 汇编结束命令

格式： END

END 是汇编语言源程序的结束标志。

3. 等值命令

格式： 字符名称 EQU 数或汇编符号

EQU 用于给其左边的“字符名称”赋值。

4. 数据地址赋值命令

格式： 字符名称 DATA 表达式

用于把表达式的值给 DATA 左边的“字符名称”赋值，表达式可以是一个数据也可以是地址。

5. 定义字节指令

格式： ［标号:］ DB 8 位二进制数表

用于把 DB 右边的二进制数表中的数据依次存放在以标号为始地址的 ROM 存储单元中。例如在 ROM 中自 2000H 单元开始存放数字 0～15 的 ASCII 码表：

```
ORG 2000H
TABLE：DB 30H，31H，32H，33H，34H，35H，36H，37H，38H，39H
       DB 41H，42H，43H，44H，45H，46H
```

再例如：DB “how are you？”定义一个数表，其中存放语句：how are you？。

6. 定义字命令

格式： ［标号:］ DW 16 位二进制数表

7. 定义空间命令

格式:［标号:］ DS 表达式

DS 用于自标号开始定义一个保留存储单元空间，表达式（或数字）为该空间的大小。

8. 位地址符号命令

格式：字符名称 BIT 位地址

BIT 用于将位地址赋给字符名称。例如：K1 BIT P1.0

将 P1.0 的位地址赋给字符名称 K1，在其后的程序段中可以用 K1 代替 P1.0，增强程序

的可读性。

3.9　如何设计键盘驱动程序实现按键的键值计算

3.9.1　编程实现键值识别

本章提出的键盘控制显示任务如图 3-8 的电路所示，实现按键识别的程序如下所示。

```
        ORG  1000H
        X  EQU 0F0H
SCAN:   MOV   A，#0FFH        ；设置 P1 端口为输入状态
        MOV   P1，A
        MOV   A，P1           ；读取 P1 端口引脚的外部电平状态
        ANL A，X              ；屏蔽电平状态低 4 位，保留高 4 位表示键值
        MOV   R0，A           ；将（A）值作为键值保存，若有键被按下以位值 0 表示
        XRL A，X              ；若无键按下，（A）=X，则将 A 清零，否则（A）≠0
        JZ      A，SCAN       ；判断有无按键按下，无则继续识别，有则等待
        SJMP    $             ；识别到按键后等待，键值在 R0 中
        END
```

这段程序利用 EQU 等值指令设定掩码 X 的值为 F0H，通过逻辑与运算实现对从 P1 端口读入数据保留高 4 位，通过逻辑异或运算判断有无按键按下。按键识别后键值在 R0 中存放，以位值 0 表示与位对应的按键按下。

3.9.2　键盘控制 LED 灯显示

键盘控制 LED 灯显示需要识别在独立式键盘键值的基础上做进一步的修改。其一，按照亮灯要求将键值转换为显示码；其二，将显示码从 LED 驱动接口输出，实现控制显示。

1.　如何将键值转换为显示码

根据要求，在图 3-8 中的 S1～S4 任意一个按键按下时点亮 LED1～LED4 中对应的一个，由电路可知若使灯点亮需要驱动 LED 的 I/O 端口输出低电平，由于键值识别程序执行得到的键值是以 0 表示按键按下的，没有按键按下的位值为 1，所以这个任务可以直接将键值作为显示驱动码使用。4 个按键与 P1 端口高 4 位连接，4 个 LED 与 P1 端口低 4 位连接，因此，作半字节交换即可实现。

2.　执行程序清单

```
        X  EQU 0F0H           ；伪指令，掩码 X 值等于 F0H
        ORG  0000H            ；伪指令，定位系统上电入口指令
        LJMP MAIN
        ORG  0030H            ；伪指令，主程序避开 ROM 的系统使用区开始存放
```

```
MAIN:
        LCALL SCAN          ；扫描识别按键
        LCALL DISP          ；显示驱动
        LCALL DELAY         ；延时
        SJMP MAIN           ；跳回 MAIN 标号实现无限循环
        ORG  0100H          ；伪指令，子程序存放区定位
SCAN:   MOV   A，#0FFH      ；设置 P1 端口为输入状态
        MOV   P1，A
        MOV   A，P1         ；读取 P1 端口引脚的外部电平状态
        ANL A，X            ；屏蔽电平状态低 4 位，保留高 4 位表示键值
        MOV   R0，A         ；将（A）值作为键值保存，若有键被按下以位值 0 表示
        XRL A，X            ；若无键按下，（A）=X，则将 A 清零，否则（A）≠0
        JZ     A，SCAN      ；判断有无按键按下，无则继续识别，有则等待
        RET                 ；识别到按键后返回，键值在 R0 中
DISP:   MOV   A，R0         ；取回键值
        SWAP   A            ；将键值交换到低半字节作为显示码
        MOV    P1，A        ；从 P1 端口输出显示码驱动 LED 灯
        RET
DELAY:  MOV   R6，#64H
LOOP1:  MOV   R7，#0F8H
        NOP
LOOP2:  DJNZ   R7，LOOP2
        DJNZ   R6，LOOP1
        RET
END                         ；伪指令，汇编结束命令
```

温馨提示：

在独立式键盘控制显示程序中，CPU 每个循环周期执行三个任务：按键识别扫描、显示驱动、延时。为何要使用延时子程序？由于在扫描程序中需要将准双向 I/O 端口的 P1 置为输入状态，这将熄灭已点亮的 LED 灯，若无延时程序将使点亮的时间极为短暂，人眼难以觉察。

考考你自己：

1. 为何要对按键消除抖动？
2. 行列式键盘的按键识别过程中，列扫描法有哪些步骤？
3. 如何设计高灵敏度、高效率的键盘接口电路?
4. 带进位加法、带借位减法运算指令适用于什么场合？
5. 图 3-8 所示的独立式键盘控制显示电路中，若要求当按键按下后对应的指示灯一经点亮就要保持直到下一个按键按下时才可熄灭，请编写程序并调试。
6. 试根据键值计算公式设计程序，对图 3-9 所示电路进行按键识别。

项目四　报警器设计

——中断原理及应用

教学目的

掌握：安全防范探测、报警电路的设计方法。
理解：中断系统的工作原理；中断初始化及中断服务程序的设计方法。
了解：安全防范电子系统的结构。

愿你知多点：

随着经济的发展和生活质量的日益提高，人们对家庭与生命财产安全越来越重视，采取了许多措施来保护家庭的安全。以往的做法是安装防盗门、防盗网，但这些也存在有碍美观、不符合防火要求、不能有效地防止坏人入侵的问题。现在电子信息技术的发展使安居工程的实现成为可能，因而家庭电子安全防范报警系统也就应运而生。这些家庭安全防范报警系统一般在案情发生时，通过对射式红外线、热释电被动式红外线、门磁等探测器进行探测，将入侵的情况转换为开关信号送入报警控制器，实现报警的功能。

在这一章中，我们将通过完成报警器设计这一任务来学习中断的基本知识。图 4-1 所示为安全防范报警器应用实例。

图 4-1　安全防范报警器应用实例

4.1　能力培养

本章通过完成报警器设计这一任务，可以培养以下能力。

1. 能设计安全防范探测与报警电路。
2. 能正确使用 MCS—51 系列单片机中断系统。
3. 能制作安全防范报警器。

4.2 任务分析

要完成此项任务，需要掌握以下 3 方面知识。

1. 如何使用常见的安全防范探测器。
2. 如何设计安全防范报警电路及其与单片机接口电路。
3. 如何设计安全防范报警程序。

4.3 如何使用 MCS—51 系列单片机中断系统

第 3 章制作了一个键盘控制显示电路，这个项目可实现当有按键按下时使对应的指示灯点亮的目的，但要实现该功能，CPU 必须一直处于按键识别状态，这样就会造成资源浪费。在图 3-10 中，将按键电平作为 4 输入与门的输入端电压，当按下按键时，$\overline{\text{INT0}}$ 端电平为低电平，则需要不断执行以下程序段检测。

```
LOOP：  JB    P3.2，LOOP      ；查询 P3.2 脚（INT0）是否为低电平
        ACALL  SCAN           ；若为低电平，则调用列扫描子程序识别按键
```

对于一个单片机测控应用系统，通常会在一个 CPU 的循环周期内安排较多的任务，那么怎样才能使 CPU 从繁重的按键识别任务中解放出来呢？对于这个问题，可以运用单片机中断系统解决。

4.3.1 中断的概念与功能

中断是指单片机在执行程序的过程中，由于单片机内部或外部的某种原因使其暂时中止原程序的执行，转而去为突发事件服务，在对突发事件处理完成后再返回原程序继续执行的过程。引起中断的事件或发出中断申请的来源称为中断源。中断源有外部 I/O 设备、定时器、串行通信以及系统故障（如掉电）、程序执行错误（如除数为 0）等。

中断执行相当于调用子程序，因而将中断处理程序称为中断服务子程序。与子程序调用的区别在于中断是随机发生的，其对中断服务子程序的启动是在检测到中断请求信号后自动完成的，而子程序的调用是由编程人员事先安排好的。因此中断又可定义为 CPU 自动执行中断服务程序并返回原程序执行的过程。

在单片机中引入中断有两个优点。第一，可以提高 CPU 的工作效率。单片机有了中断功能后，CPU 和外设、定时器、通信等部件就可以同步工作，CPU 只要启动这些部件就可以执行原程序，而这些部件在完成指定的操作后向 CPU 发出中断请求，CPU 暂时中止原程序的执行转去为这些部件服务，完成任务后继续执行原程序。在中断服务程序中，CPU 向这些部件下达新的命令或传送新的数据，之后这些部件就可以继续与 CPU 并行工作。第

二，便于实时处理。有了中断功能，就可以实时测控现场的各个参数、信息，在任何时刻都可以向 CPU 发出中断申请，并会受到及时处理。

作为单片机的中断系统，包括硬件电路和软件程序，主要实现几大功能。第一，中断响应。识别中断标志信号并由 CPU 决定是否响应，一旦作出响应则应保护原程序端点与现场，并转到中断服务程序的入口。第二，中断返回。CPU 在执行完中断服务程序并遇到中断返回指令时，自动取出堆栈中的断点地址以返回到原程序断点处继续执行原程序。第三，中断优先级排队与中断嵌套。如同接电话与开水壶两个事件相比开水壶的处理更为紧急，即中断系统应能优先响应中断优先级高的部件，或者在处理低优先级事件时能够被高优先级中断，嵌套进入更高一级的中断服务，当返回时由高级返回低级中断，处理完毕再返回原程序执行。

温馨提示：

我们可以想象，当你在家看书时电话铃响了，你将如何行动呢？首先，将书的页码记下，然后去接电话，讲完电话返回来再找回以前的页码继续看书。若是在接电话时烧开水的壶沸腾了怎么办呢?势必是中止接电话去处理开水壶，处理完再回来继续来接电话，直到挂机后再返回继续看书。单片机中断系统的概念与此十分类似。图 4-2 所示为看书—电话中断与 CPU 中断对比示意图。

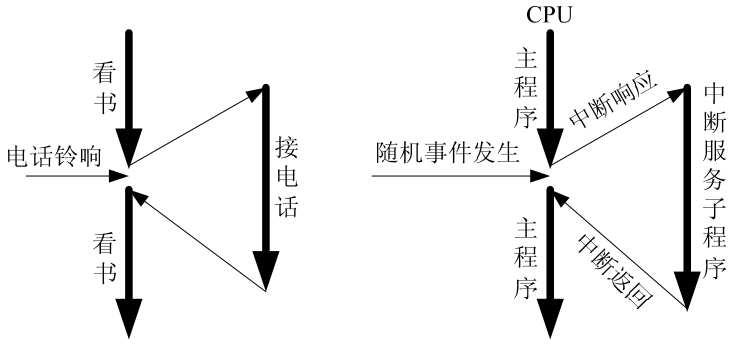


图 4-2　看书—电话中断与 CPU 中断对比示意图

4.3.2　MCS—51 系列单片机的中断系统

MCS—51 系列单片机有 5 个中断源，包括 2 个外部中断源、2 个定时/计数器中断源和 1 个串行口中断源。中断系统操作过程示意图如图 4-3 所示，由中断源置位特殊功能寄存器 TCON 中相对应的位，在中断允许寄存器 IE 的允许位值为 1 的前提下将中断信号送入中断响应队列引起 CPU 响应中断，中断队列分高优先级和低优先级两个，由 IP 寄存器中相应的位值决定。

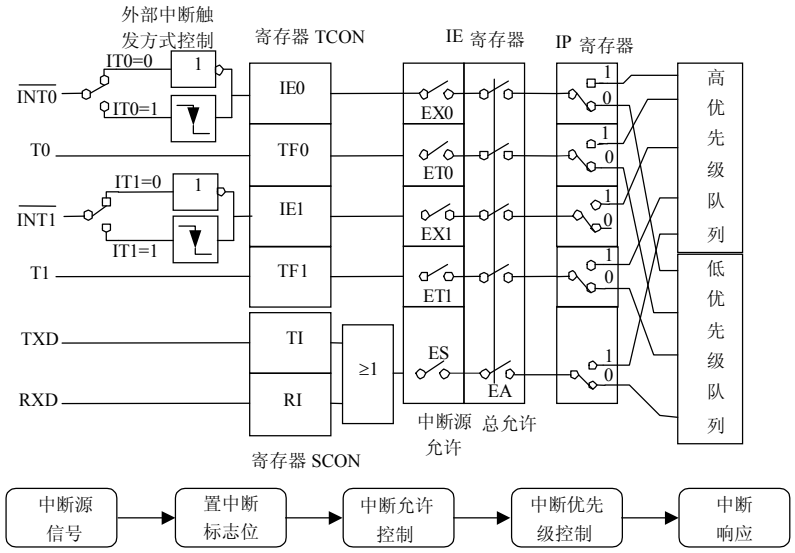


图 4-3　中断系统操作过程示意图

1. 中断源

外部中断源 0：即 $\overline{\text{INT0}}$ 端，其中断请求信号由 P3.2 处输入。

外部中断源 1：即 $\overline{\text{INT1}}$ 端，其中断请求信号由 P3.3 处输入。外部中断请求有两种信号方式，即电平触发方式和脉冲信号下降沿触发方式。在电平触发方式下，CPU 在每个机器周期的 S5P2 时刻都从 $\overline{\text{INT0}}$（P3.2）/ $\overline{\text{INT1}}$（P3.3）端输入的电平处采样，若采样到低电平，则判定为中断请求，置位中断标志位 IE0（TCON.1）、IE1（TCON.3），在中断允许寄存器的 EX0（IE.0）、EX1（IE.2）、EA（IE.7）为 1 的情况下即可进入中断服务子程序；在脉冲信号下降沿触发方式下，CPU 在每个机器周期的 S5P2 时刻都从 $\overline{\text{INT0}}$（P3.2）/ $\overline{\text{INT1}}$（P3.3）端输入的电平处采样，若在两次采样中，前一个机器周期采样信号为高电平，后一个机器周期采样信号为低电平，即采样到一个下降沿，则判定为一个中断请求信号，将相应的中断标志位置 1 请求中断。$\overline{\text{INT0}}$ 和 $\overline{\text{INT1}}$ 的中断触发方式由 IT0（TCON.0）、IT1（TCON.2）设置。

定时器/计数器中断：包括定时器/计数器 0（即 T0）和定时器/计数器 1（即 T1），当作为定时器使用时，其中断请求信号取自单片机内部定时脉冲信号，当作为计数器使用时，其中断请求信号取自 T0（P3.4）或 T1（P3.5）引脚。启动定时器后，每个机器周期会在其引脚上每检测到一个脉冲信号信号 T，这时计数器就加 1 一次，当溢出时即刻置位中断标志位 TF0（TCON.5）/TF1（TCON.7）。

串行口中断：串行口中断分为发送中断与接收中断两种，每当串行口发送端（TXD，P3.1 脚）发送或从接收端（RXD，P3.0 脚）接收完一组串行数据时，就产生一个中断信号将中断标志位 TI（SCON.1）或 RI（SCON.0）置 1。

MCS—51 单片机中断系统有关的寄存器的位分配见表 4-1 所列。

表 4-1　MCS—51 单片机中断系统有关的寄存器的位分配

TCON 寄存器位功能	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
	T1 中断标志	T1 启动	T0 标志	T0 启动	外部中断 1 标志	外部中断 1 触发方式	外部中断 0 标志	外部中断 0 触发方式
SCON 寄存器位功能	SM0	SM1	SM2	REN	TB8	RB8	TI	RI
	串行口方式位	串行口方式位	多机方式位	接收允许	发送第九位	接收第九位	接收中断标志	发送中断标志
IE 寄存器位功能	EA	—	—	ES	ET1	EX1	ET0	EX0
	总允许	未定义	未定义	串口中断允许	T1 中断允许	外部中断 1 允许	T0 中断允许	外部中断 0 允许
IP 寄存器位功能	—	—	—	PS	PT1	PX1	PT0	PX0
	未定义	未定义	未定义	串行口中断优先级	T1 中断优先级	外部中断 1 优先级	T0 中断优先级	外部中断 0 优先级

温馨提示：

特殊功能寄存器 TCON、SCON 分别属于定时/计数器电路、串行端口通信电路部件，属于控制寄存器，定时/计数器还有工作方式寄存器 TMOD 和加计数器 TH0、TL0、TH1、TL1，串行端口有发送/接收缓冲器 SBUF。

2. 中断允许

MCS—51 系列单片机中断系统应用时，需要程序设计者通过编程设置中断允许位开启或禁止各个中断源，图 4-4 所示为中断允许示意图。

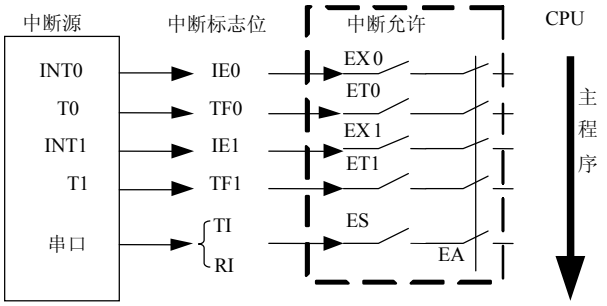


图 4-4　中断允许示意图

3. 中断优先级与中断嵌套

为了更好地实现中断控制，中断系统优先级依据的控制原则为低优先级中断不能打断高优先级中断的服务程序，但高优先级可以打断低优先级的中断服务，称为中断嵌套，其示意图如图 4-5 所示。

同级中断不能打断同级中断服务，如果多个同级中断源同时申请中断时，CPU 按照默认顺序响应，即按外部中断 0→定时/计数器 0→外部中断 1→定时/计数器 1→串行中断优先级顺序响应。

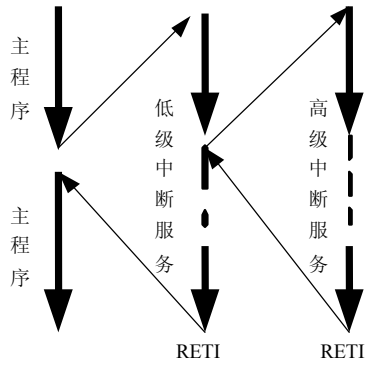


图 4-5　中断嵌套示意图

4. 中断响应过程与中断源入口

中断响应过程包括保护断点和将程序转向中断服务程序入口地址。首先中断系统通过硬件自动生成一个长调用指令（LCALL），将断点地址压入堆栈保护（不保护累加器 A、PSW 和其他寄存器内容）。然后将对应的中断入口矢量地址送入程序计数器 PC，使程序转向该中断入口执行中断服务程序。MCS—51 单片机各中断入口地址由硬件实现设定，其中断入口地址见表 4-2 所列。

表 4-2　中断入口地址

中 断 源	入口地址
外部中断 $\overline{\text{INT0}}$	0003H
定时/计数器 T0 中断	000BH
外部中断 $\overline{\text{INT1}}$	0013H
定时/计数器 T1 中断	001BH
串行口中断	0023H

温馨提示：

入口地址是 ROM 单元地址，两个中断源入口单元之间相隔 8 个字节单元。中断响应后转向中断向量入口执行中断服务子程序，然而一般的中断服务子程序都超过 8 个字节，因此通常是使用 ORG 伪指令把中断服务子程序放在 ROM 的另外一个区域，而在入口地址处单元内存放跳转指令 LJMP，无条件地转到中断服务子程序区。

4.3.3　中断编程

如前所述，中断系统的主要作用是实现多个任务分时操作和紧急任务实时处理，即CPU 可以和多个外设同时工作，且 CPU 能够及时处理随机事件。通常将单片机测控应用系统的功能拆分为若干个子任务，将实时性要求低的子任务作为无限循环序列由 CPU 不断执行，而将实时性高、能够由其他电路部件完成的子任务设置为中断服务子程序，当中断发生时申请中断响应执行，这种运行模式称为单片机前后台运行模式，其示意图如图 4-6 所示。

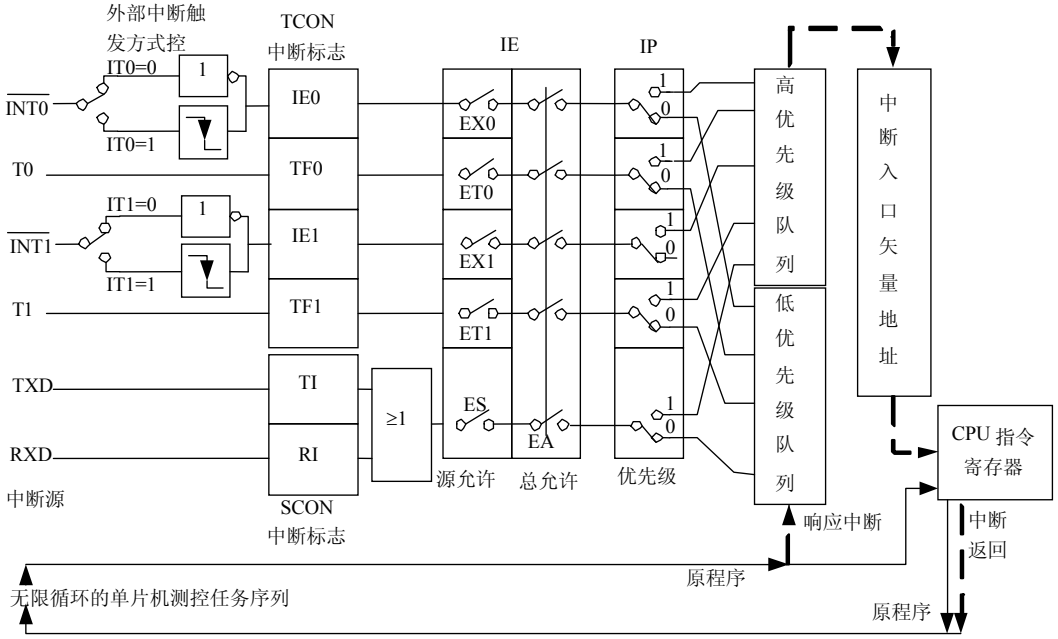


图 4-6　单片机前后台运行模式示意图

单片机的这种运行模式需要对程序作出特殊处理，主要是合理地对程序存储器 ROM 进行分区，在系统入口处存放主程序（由中断初始化和无限循环子任务序列组成）跳转指令，在中断入口地址单元存放中断服务子程序跳转指令，在较高的地址单元之后开始存放主程序和中断服务子程序。另外，在中断服务子程序开始处做好保护现场和在中断返回之前做好恢复现场操作。ROM 分区与程序结构模式如图 4-7 所示。

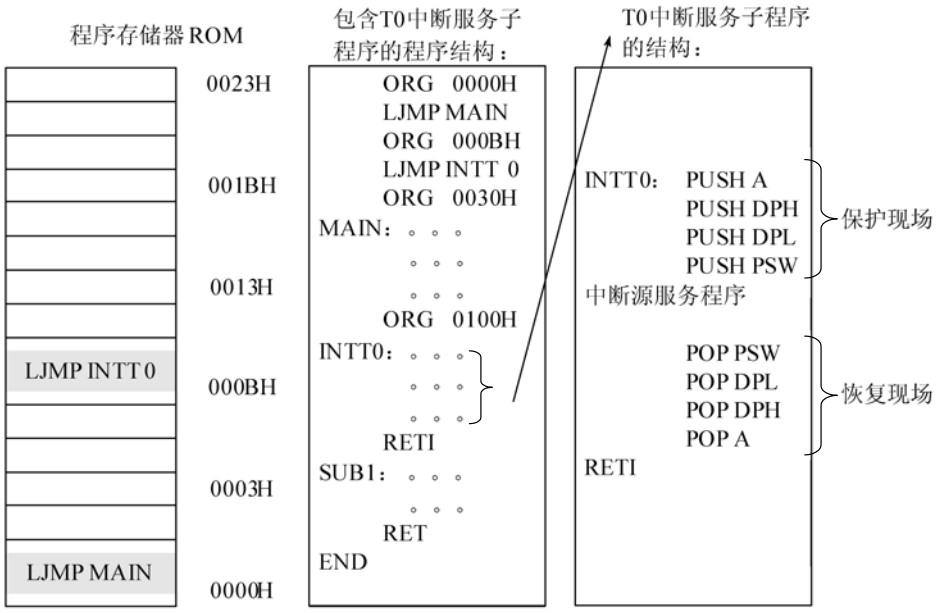


图 4-7　ROM 分区与程序结构模式示意图

例 4-1　编写中断系统初始化编程，要求外中断 1、定时器 1 允许中断，其他不允许。

本例主要要求对中断允许特殊功能寄存器 IE 的相应值进行设置，既可以用字节操作指令实现，也可以用位指令实现，方法如下所示。

方法 1 是字节操作指令：

```
MOV  IE，＃8CH        或  MOV  A8H，＃8CH
```

方法 2 是位操作指令：

```
SETB  EA              ；使 EA＝1，CPU 开中断
SETB  ET1             ；使 ET1＝1，定时/计数器 1 允许中断
SETB  EX1             ；使 EX1＝1，外中断 T1 允许中断
```

例 4-2　编写中断系统初始化编程，要求将 T0、外中断 1 设为高优先级，其他均设为低优先级。

IP 的首 3 位没有使用，可取任意值（若设为 000），后面根据要求为 06H，指令如下所示。

```
MOV   IP，#06H
```

例 4-3　编写中断系统初始化编程，要求将外部中断 1 设置为低电平触发、高优先级，方法如下所示。

方法 1 是位操作指令的程序：

```
SETB  EA
SETB  EX1  ;      开外中断 1 中断
SETB  PX1  ;      令外中断 1 为高优先级
CLR     IT1
```

方法 2 是字节型指令的程序：

```
MOV   IE，＃84H       ；开外中断 1 中断
ORL   IP，＃04H       ；令外中断 1 为高优先级
ANL   TCON，＃0FBH    ；令外中断 1 为电平触发
```

例 4-4　图 4-8 所示为蒸汽锅炉越限报警系统，对液位、压力、温度等物理量进行监测，实现越限告警。其中，液位上限 SL1 和下限 SL2 开关取自“色带指示报警仪”，分别接 P1.3，P1.2。蒸汽压力下限 SP 开关取自“压力计”，接 P1.1。炉膛温度上限 ST 开关取自“动圈式温度计”，接 P1.0。P1.7～P1.4 输出接发光二极管，与 4 个参数对应，请编程实现越限时将相应的 LED 灯点亮。

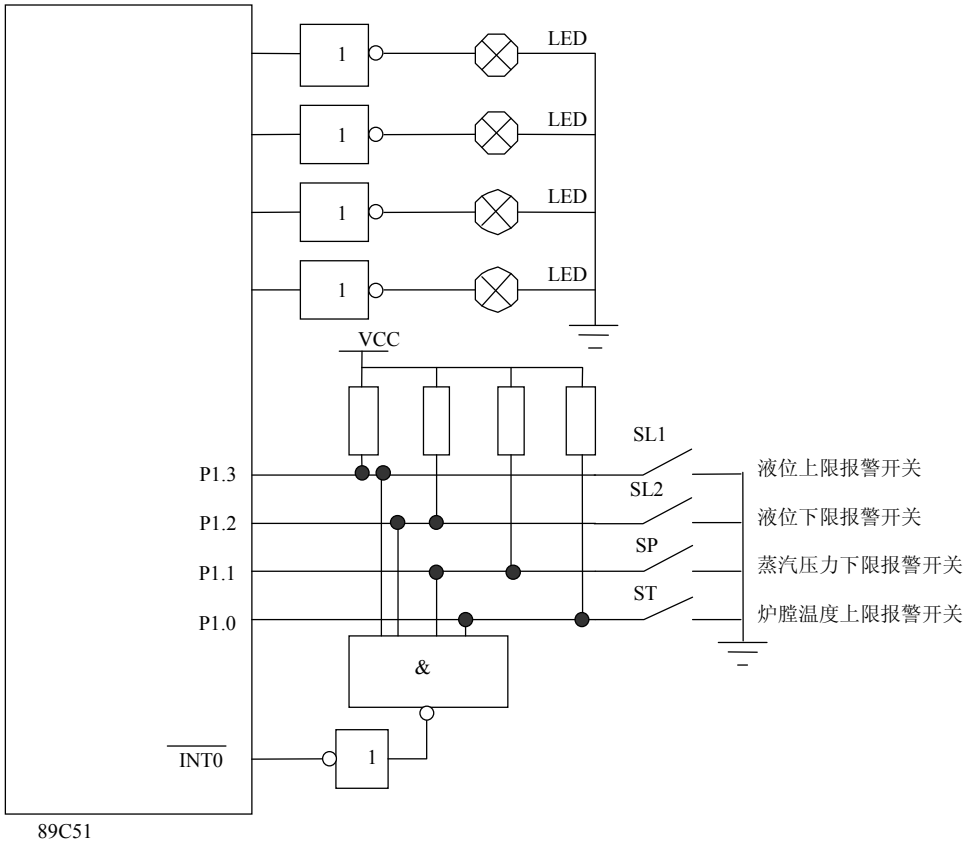


图 4-8　蒸汽锅炉越限报警系统

程序清单如下：

```
        ORG   0000H
        AJMP  MAIN          ；上电自动转主程序
        ORG   0003H
        AJMP  ALARM         ；转中断服务程序
        ORG   0200H
MAIN:   SETB  IT0           ；置为边沿触发
        SETB  EX0           ；允许中断
        SETB  EA            ；CPU 开中断
HERE:   SJMP  HERE          ；等待中断
        ORG   0210H
ALARM:  MOV  A，＃0FFH      ；高 P1 口为输入口
        MOV  P1，A
        MOV  A，P1          ；取消报警状态
        SWAP  A             ；（P1.7～P1.4）（P1.3～P1.0）
        MOV  P1，A          ；输出报警信号
        RETI
```

温馨提示：

CPU 响应某中断请求后，在中断返回前，应该撤除该中断请求，否则会引起另一次中断。

- 边沿激活的外部中断：CPU 在响应中断后，也是用硬件自动清除有关的中断请求标志 IE0 或 IE1。
- 电平触发外部中断撤除方法较复杂。因为在电平触发方式中，CPU 响应中断时不会自动清除 IE1 或 IE0 标志，所以在响应中断后应立即撤除 INT0 或 INT1 引脚上的低电平。
- 定时器 0 或 1 溢出：CPU 在响应中断后，硬件清除了有关的中断请求标志 TFO 或 TF1，即中断请求是自动撤除的。
- 在硬件上，CPU 对 INT0 和 INT1 引脚的信号不能控制，所以这个问题要通过硬件来解决，同时还要有软件的配合。
- 串行口中断：CPU 响应中断后，没有用硬件清除 T1、R1，故这些中断不能自动撤除，而要靠软件来清除相应的标志。

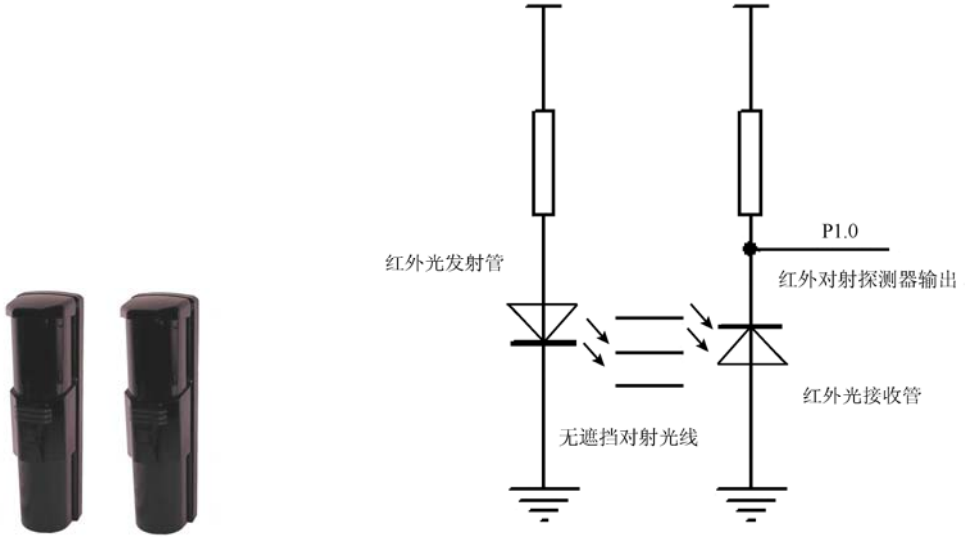
4.4　如何设计安全防范报警电路及其与单片机接口电路

安全防范报警电路包括警情探测电路、报警电路和报警控制器三个部分。警情探测器电路将入侵转换为电平信号，并将该电平信号作为由单片机构成的报警控制器的外部中断输入信号，经单片机程序判断后输出信号，再启动报警电路工作。

4.4.1　如何使用安全防范探测器

随着电子技术的发展，用于安全防范报警系统的探测器种类越来越多，按其发送与接收方式可以分为有线和无线等形式，如红外线对射式探测器、人体热释电红外线感应式探测器、门磁开关、金属网断线式以及超声波探测式、微波探测式等。下面介绍运用红外线对射式探测器、人体热释电红外线感应式探测器、门磁开关、金属网断线式安全防范探测器设计的报警器，并对其电路结构和工作原理进行分析。

红外线对射式探测器的实物及电路结构如图 4-9 所示，它由红外线光管和红外线接收管组成。当发射、接收之间无遮挡时，在光电流作用下导通电阻值极小，探测器输出点为低电平；当有物体遮挡时，在暗电流作用下导通电阻值变得很大，输出高电平。

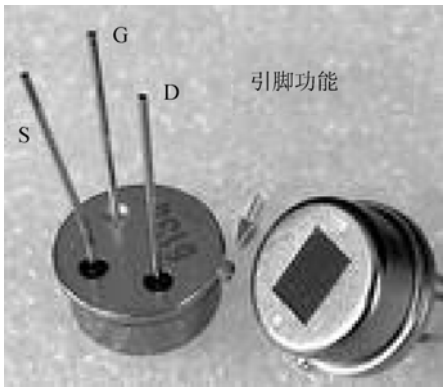


（a）实物　　（b）电路结构

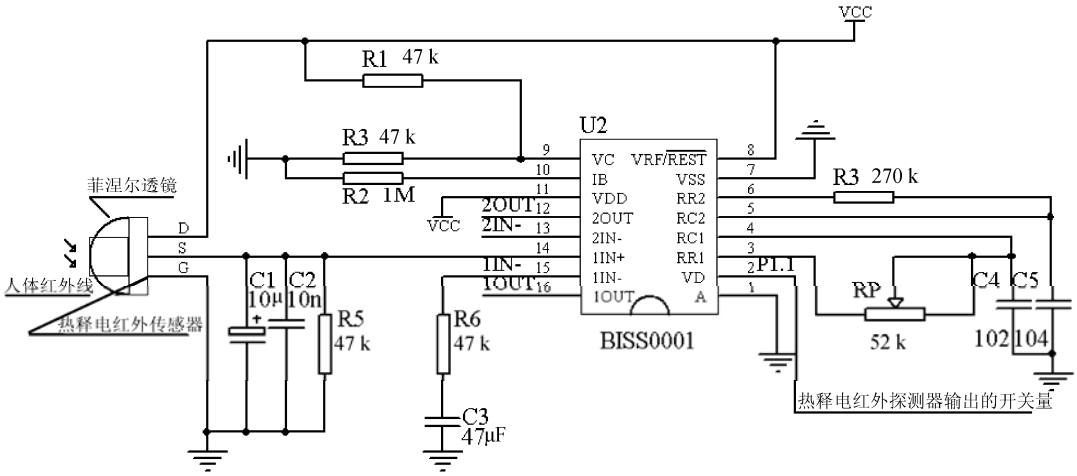
图 4-9　红外线对射式探测器的实物及电路结构

人体热释电红外线感应式探测器的实物及电路结构如图 4-10 所示。

（b）实物



（b）实物



（c）电路结构

图 4-10　人体热释电红外线感应式探测器的实物及电路结构

人体热释电红外线感应式探测器电路由热释电传感器、红外信号放大检测电路和菲涅耳透镜组成，当菲涅耳透镜前面形成一个半球形的探测区域时，且区域内无人移动（无红外辐射热源）时，探测器输出点为低电平，当有人移动时输出高电平。

门磁开关和金属网断线式探测器的电路结构如图 4-11 所示，其工作原理与第 3 章所述按键相同。当门或窗闭合时门磁靠近开关建立磁场使其闭合，输出低电平，否则输出变为高电平。金属网断线式探测器的金属线断开时相当于常闭开关打开将输出高电平。

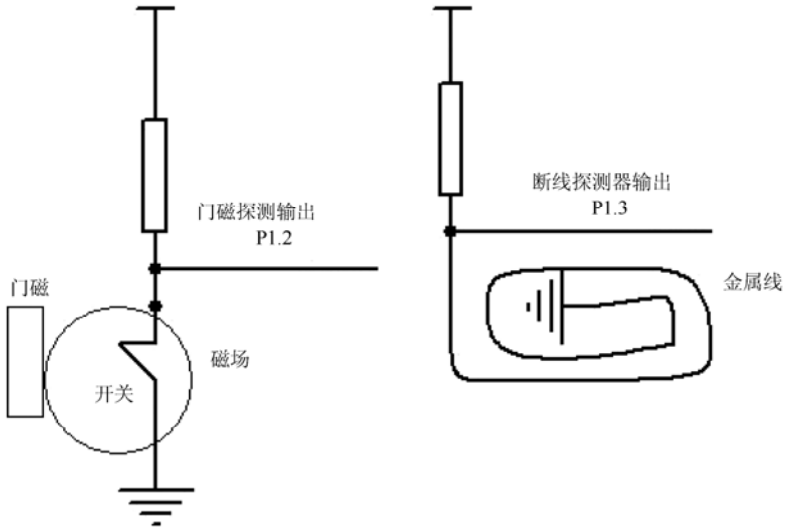


图 4-11　门磁开关和金属网断线式探测器的电路结构

4.4.2　安全防范报警电路

安全防范报警电路如图 4-12 所示。安全防范报警电路由振荡电路、输出驱动 LED 和扬声器组成。图中 IC1（555 电路）与 R5、R6、C3 组成频率固定的低频振荡器，其输出驱动 LED 灯闪烁；IC2（555 电路）与 R8，R9，C4 组成频率可变的振荡器，其参考电压取自充电电容器 C3 的端电压，并经 PNP 三极管耦合得到，因而 IC2 的振荡频率会产生变化，IC2 的 PIN5 和 PIN3 的波形图如图 4-13 示。

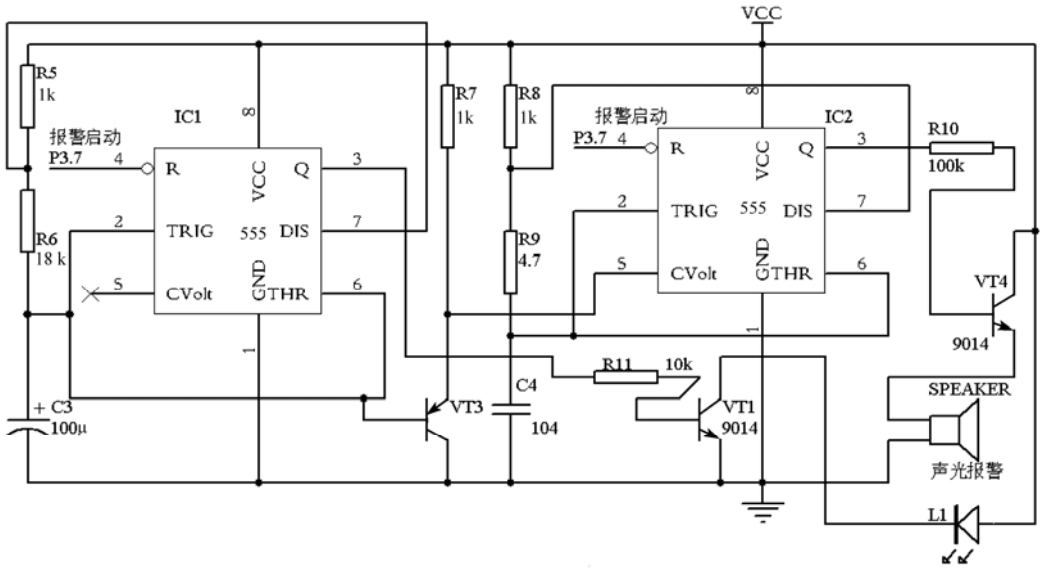


图 4-12　安全防范报警电路

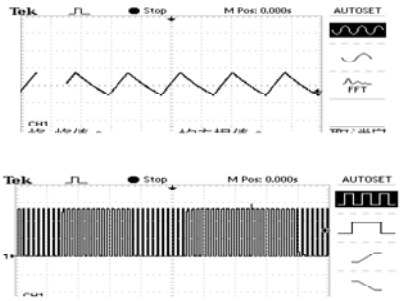


图 4-13　IC2 的 PIN5 和 PIN3 的波形图

其中上半部分波形为充电电容器 C3 的端电压，下半部分波形为输出到扬声器的电压波形。IC1 和 IC2 的复位端由单片机 I/O 端口控制是否振荡，一旦起振，扬声器发出警笛的鸣响，对犯罪分子极具威慑作用。

4.4.3　安全防范报警电路与单片机接口电路

安全防范报警电路要求能够灵敏可靠工作，一旦有人入侵布防区域立即启动声光报警电路及时通知保安或警醒居民，同时可吓阻犯罪分子的侵入。为此，将警情探测电路接入单片机的外部中断输入端由硬件电路实现警情识别，将启动报警的子任务作为中断服务子程序。由于报警器设置了 4 路探测信号输入，且都是高电平信号报警，因此通过 7425 双路四输入或非门将报警信号送入 $\overline{\text{INT0}}$ 端，同时将 4 路信号接入普通 I/O 端口的 P1.0～P1.3，用于扩展防区显示电路，限于篇幅本任务略去。单片机 I/O 端口 P2.0 驱动的 LED 灯用于指示防区平安和系统正常工作。另外，设置系统工作开关，其输出接入 P3.6。当开关接通时，P3.6 变为低电平系统开始工作，开关断开时系统停止工作，并且可用作为警报解除开关。安全防范报警电路与单片机接口电路见图 4-14。

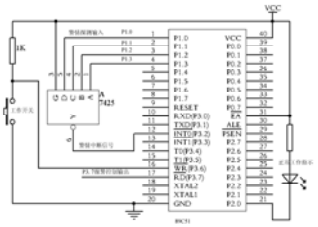


图 4-14　安全防范报警电路与单片机接口电路

4.5　如何设计安防报警程序

4.5.1　系统初始化及中断服务程序

由于警情探测电路无入侵情况下均输出低电平，有入侵时输出高电平，并且警情中断信号（低电平）不能自行消除，所以外部中断 0，设置为脉冲信号下降沿触发方式。另外

以寄存器 F0 记录平安与入侵状态，0 表示平安状态，1 表示入侵状态。初始化程序如下所示。

```
SETB IT0                ；设置外部中断 0 下降沿触发
SBTB EX0                ；允许外部中断 0
SETB IP0                ；设置外部中断 0 高优先级
SETB P2.0               ；点亮系统工作/防区安全指示灯
CLR  P3.7               ；系统初始阶段，关闭警报
CLR F0                  ；系统初始阶段，清零入侵状态寄存器
```

当检测到防区入侵后进行报警，中断服务子程序需完成置位入侵标志位、启动报警电路和关闭平安指示 LED 灯，程序如下所示。

```
INT0：CLR P2.0          ；防区有入侵，关闭防区平安指示灯
      SETB F0           ；防区有入侵，置位入侵状态寄存器
      SETB P3.7         ；防区有入侵，开启警报
      LCALL DELAY       ；延时
      RETI              ；中断返回
```

4.5.2 主程序

主程序是一个无限循环的子任务序列，主要完成的子任务有两个，一个为检测系统工作状态启动，另一个为关闭报警器和驱动防区安全指示灯闪烁。程序如下所示。

```
LOOP:   SETB P3.6          ; P3.6 端设置为输入状态
        JNB P3.6, RUN      ; 检测系统工作状态开关是否导通
        CLR EA             ; 系统关闭，禁止警情中断
        CLR P3.7           ; 系统关闭，关闭警报
        CLR P2.0           ; 系统关闭，关闭防区安全正常工作指示灯
        SJMP LOOP          ; 跳转至子任务序列首部
RUN:    JB F0, ALARM       ; 系统工作，检测是否入侵状态
        SETB EA            ; 系统工作，防区无入侵开启警情中断
        CPL P2.0           ; 系统工作，防区无入侵驱动防区平安指示灯闪烁
        LCALL DELAY        ; 延时
ALARM:  CLR P2.0           ; 系统工作，防区有入侵，关闭防区平安指示灯
        SETB P3.7          ; 系统工作，防区有入侵，开启警报
        SJMP LOOP          ; 跳转至子任务序列首部
```

4.5.3 程序清单

```
        ORG 0000H
        LJMP MAIN
        ORG 0030H
        LJMP INT0
        ORG 0030H
MAIN:   SETB IT0           ; 设置外部中断 0 下降沿触发
        SBTB EX0           ; 允许外部中断 0
        SETB IP0           ; 设置外部中断 0 高优先级
        SETB P2.0          ; 点亮系统工作/防区安全指示灯
        CLR  P3.7          ; 系统初始阶段，关闭警报
        CLR F0             ; 系统初始阶段，清零入侵状态寄存器
LOOP:   SETB P3.6          ; P3.6 设置为输入状态
        JNB P3.6, RUN      ; 检测系统工作状态开关是否导通
        CLR EA             ; 系统关闭，禁止警情中断
        CLR P3.7           ; 系统关闭，关闭警报
        CLR P2.0           ; 系统关闭，关闭防区安全正常工作指示灯
        SJMP LOOP          ; 跳转至子任务序列首部
RUN:    JB F0, ALARM       ; 系统工作，检测是否入侵状态
        SETB EA            ; 系统工作，防区无入侵开启警情中断
        CPL P2.0           ; 系统工作，防区无入侵驱动防区平安指示灯闪烁
        LCALL DELAY        ; 延时
```

```
ALARM:  CLR P2.0            ；系统工作，防区有入侵，关闭防区平安指示灯
        SETB P3.7           ；系统工作，防区有入侵，开启警报
        SJMP LOOP           ；跳转至子任务序列首部
        ORG  0100H
INT0:   CLR P2.0            ；防区有入侵，关闭防区平安指示灯
        SETB F0             ；防区有入侵，置位入侵状态寄存器
        SETB P3.7           ；防区有入侵，开启警报
        LCALL DELAY         ；延时
        RETI                ；中断返回
DELAY:  MOV   R6，#64H      ；延时子程序
LOOP1:  MOV   R7，#0F8H
        NOP
LOOP2:  DJNZ   R7，LOOP2
        DJNZ   R6，LOOP1
        RET
END
```

安全防范报警电路的程序流程图如图 4-15 所示。

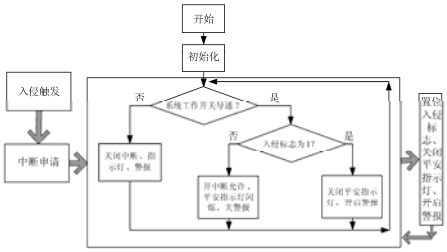
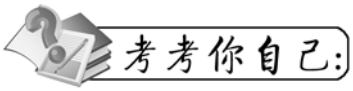


图 4-15　安全防范报警电路的程序流程图

温馨提示：

制作安防报警电路之前，首先要了解 555、7425、LHI958 及 BIS0001 等器件的内部结构及管脚排序，在对整个电路有一个完整理解的基础上，可将电路分为几个模块独立测试，通过独立测试后再进行整机联调。



1. 请简述中断的概念。
2. 按下列要求分别设置相关控制位。

（1） $\overline{\text{INT0}}$ 为边沿触发方式。

（2） $\overline{\text{INT1}}$ 为电平触发方式。

3. 根据下列已知条件，试求中断开关状态。

（1） IE=93H

（2） IE=84H

（3） IE=92H

（4） IE=17H

4. 安全防范系统有哪些类型的探测器？

项目五　定时控制器的设计

——定时/计数器原理及应用

教学目的

掌握：定时控制器的设计方法。
理解：单片机定时/计数器的编程方法。
了解：定时/计数器的结构。

愿你知多点：

在我们的日常生活中，经常使用各种定时/计数器以及定时控制产品，如闹钟、秒表、计费器、洗衣机洗衣时间的控制、微波炉烧烤时间的控制等，那么这些定时/计数器是如何制作的呢？在这一章中，我们将通过完成定时控制器的设计这一任务来学习制作定时/计数器的方法及相关知识。

5.1　能力培养

本章通过完成定时控制器这一任务，可以培养以下能力。

1. 能正确计算定时时间。
2. 能正确设计单片机计数器。
3. 能正确设计单片机定时控制器。

5.2　任务分析

要完成此项任务，需要掌握以下两方面知识。

1. 如何使用定时 / 计数器。
2. 如何设计定时控制器。

5.3　如何使用定时/计数器

5.3.1　定时/计数器的结构

定时/计数器的结构如图 5-1 所示。

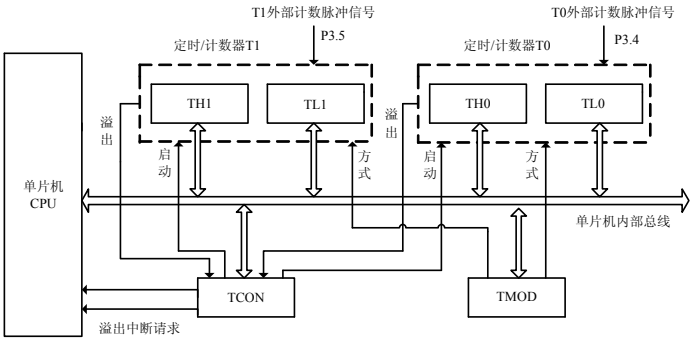


图 5-1　定时/计数器的结构

从图中可以看出，每个 16 位定时/计数器分别由两个 8 位专用寄存器组成，即 T0 由 TH0 和 TL0 构成，T1 由 TH1 和 TL1 构成。另外定时/计数器内部还有一个 8 位的定时/计数器的方式寄存器 TMOD 和一个 8 位的定时/计数器的控制寄存器 TCON，这些寄存器之间是通过内部总线和控制逻辑电路连接起来的。TMOD 主要用于选择定时/计数器的工作方式，TCON 主要是用于控制定时/计数器的启动/停止，此外 TCON 还可以保存 T0、T1 的溢出和中断标志。

定时/计数器可以工作在定时方式，也可以工作在计数方式，其中对内部脉冲信号进行的计数称为定时，对外部脉冲信号进行的计数才称为计数，当定时器工作在计数方式时，外部事件由 T0 的引脚 P3.4 和 T1 的引脚 P3.5 输入。

5.3.2　定时/计数器的结构与工作原理

当定时/计数器工作在定时方式时，即可对机器周期进行计数，每过一个机器周期，计数器加 1，直至计满溢出为止。由于一个机器周期等于 12 个振荡周期，所以计数频率 $f_{count}=1/12f_{osc}$。如果晶体振荡器的频率为 12 MHz，则计数周期为

$$T=1/（12\times10^6）\text{Hz}\times1/12=1\ \mu\text{s}$$

这是最短的定时时间。若要延长定时时间，则需要改变定时器的初值，并要适当选择定时器的长度（如 8 位、13 位、16 位等）。

当定时/计数器为计数工作方式时，通过 T0 的引脚 P3.4 和 T1 的引脚 P3.5 对外部信号进行计数。计数器在每个机器周期的 S5P2 期间由引脚输入电平处采样。若前一个机器周期采样值为 1，下一个机器周期采样值为 0，则计数器加 1。在此后的机器周期 S3P1 期间，新的计数值进入计数器。所以检测一个由 1 至 0 的跳变需要两个机器周期，故外部计数的最高计数频率为振荡频率的 1/24。若假设选用频率为 12 MHz 的晶体振荡器，则最高计数频率为 0.5 MHz。虽然对外部输入信号的占空比无特殊要求，但为了确保给定电平在变化前至少被采样一次，外部计数脉冲信号的高电平与低电平保持时间均需在一个机器周期以上。

当 CPU 用软件给定时器设置了工作方式之后，定时器就会按设定的工作方式独立运行，不再占用 CPU 的操作时间，除非定时器计满溢出，才可能中断 CPU 的当前操作。CPU 也可以重新设置定时器工作方式，以改变定时器的操作。由此可见，定时器是单片机中工

作效率高而且工作灵活的部件。

5.3.3　单片机定时/计数器的方式寄存器和控制寄存器

1.　定时/计数器的方式寄存器 TMOD（字节地址 89H）

定时/计数器的方式寄存器 TMOD 用于选择定时/计数器的工作方式，其格式如下所示。

D7	D6	D5	D4	D3	D2	D1	D0
GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0
T1				T0			

由上面的内容可以看出 TMOD 被分成两部份，每部分 4 位，分别用于控制 T1 和 T0。

（1） GATE：门控位。当 GATE=0 时，只要用软件使 TCON 中的 TR0 或 TR1 置 1，就可以启动定时/计数器；当 GATE=1 时，不但要用软件使 TCON 中的 TR0 或 TR1 置 1，还要使外部中断引脚 $\overline{INT0}$ 或 $\overline{INT1}$ 为高电平，这样才能启动定时/计数器。

（2） C/$\overline{T}$：定时/计数器功能选择。当 C/$\overline{T}$=0 时，定时/计数器工作在定时状态；当 C/$\overline{T}$=1 时，定时/计数器工作在计数状态。

（3） M1、M0：定时/计数器的工作方式选择。共 4 种工作方式，见表 5-1 所列。

表 5-1　定时/计数器工作方式选择

M1、M0	工作方式	功能说明
0　0	方式 0	13 位定时/计数器
0　1	方式 1	16 位定时/计数器
1　0	方式 2	8 位自动重装初值定时/计数器
1　1	方式 3	T0 分为两个独立的 8 位定时/计数器；T1 在该方式下停止计数。

2.　定时/计数器的控制寄存器 TCON（字节地址 88H）

定时/计数器的控制寄存器 TCON 主要用于控制定时/计数器的启动，格式如下所示。

D7	D6	D5	D4	D3	D2	D1	D0
TF1	TR1	TF0	TR0				

与定时/计数有关　　与中断有关（项目四中已经介绍）

（1） TR0：定时/计数器 T0 的启动控制位。当 TR0=1 时，启动 T0 定时（或计数）；当 TR0=0 时，T0 停止工作。

（2） TF0：定时/计数器 T0 的溢出标志。当定时/计数器 T0 的计数产生溢出时，由硬件自动将 TF0 置 1，CPU 响应中断后，由硬件自动将 TF0 清零。

（3） TR1：定时/计数器 T1 的启动控制位。含义与 TR0 类似。

（4） TF1：定时/计数器 T1 的溢出标志。含义与 TF0 类似。

5.3.4 定时/计数器的工作方式

定时/计数器 T0 有 4 种工作方式，T1 有 3 种工作方式（无工作方式 3），因为 T1 的工作方式和 T0 的前 3 种工作方式基本相同，因此以 T0 为例，介绍定时/计数器的工作方式。

1. 工作方式 0

定时/计数器的工作方式 0 也称为 13 位定时/计数器方式。它由 TL（0/1）的低 5 位和 TH（0/1）的 8 位构成，TL（0/1）的高 3 位未使用。定时/计数器工作于方式 0 的逻辑结构图如图 5-2 所示。

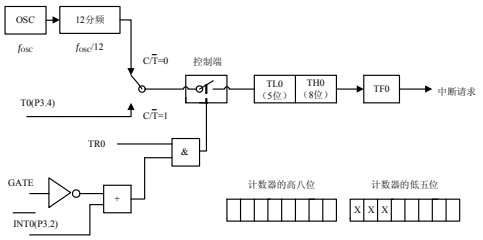


图 5-2　定时/计数器工作于方式 0 的逻辑结构图

2. 工作方式 1

工作方式 1 也被称为 16 位定时/计数器方式，它由 TL（0/1）的低 8 位和 TH（0/1）的高 8 位构成。定时/计数器工作于方式 1 的逻辑结构图如图 5-3 所示。

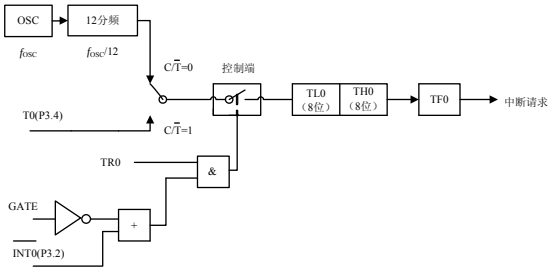


图 5-3　定时/计数器工作于方式 1 的逻辑结构图

3. 工作方式 2

工作方式 2 也被称为 8 位自动重装初值的定时/计数器方式。在工作方式 2 中，只有低 8 位参与计数，高 8 位不参与计数，用作预置数的存放。每当计数溢出时，单片机就会打开 T0（或 T1）高 8 位与低 8 位之间的开关，预置数就进入到低 8 位。通常工作方式 2 用于波特率发生器。定时/计数器工作于方式 2 的逻辑结构图如图 5-4 所示。

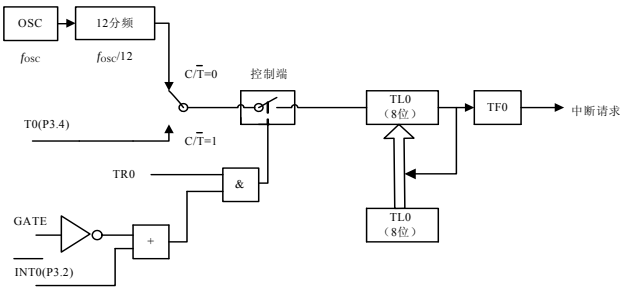


图 5-4　定时/计数器工作于方式 2 的逻辑结构图

4. 工作方式 3

在这种工作方式下，T0 被拆分成 2 个独立的定时/计数器。其中 TL0 可以构成 8 位的定时器或计数器，TH0 只能作为定时器用，因为定时/计数器使用时需要控制，溢出时需要有溢出标记，T0 被分成两个来用，那就要两套控制及溢出标记，在方式 3 中，TL0 还是用原来的 T0 的标记，TH0 则借用 T1 的标记，这样 T1 就没有了标记。一般情况下，只要 T0 工作在方式 3，T1 工作在方式 2，定时/计数器可做为串行口波特率发生器。定时/计数器工作于方式 3 中 TL0 的逻辑结构图如图 5-5 所示，TH0 的逻辑结构图如图 5-6 所示。

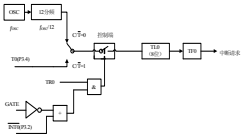


图 5-5　在定时/计数器工作于方式 3 中 TL0 的逻辑结构图

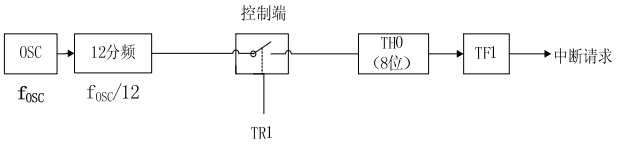


图 5-6　在定时/计数器工作于方式 3 中 TH0 的逻辑结构图

5.3.5　定时/计数器的定时/计数范围

1.　工作方式 0

工作方式 0 为 13 位的定时/计数器工作方式，因此最多可以计到 2^{13}，即 8192 次。如果单片机的晶体振荡器频率为 6 MHz，则最长的定时时间为

$$t_d=2^{13}\times T_0=2^{13}\times\frac{1}{f}=8192\times 1\ \mu s=8192\ \mu s$$

2.　工作方式 1

工作方式 1 为 16 位的定时/计数器工作方式，因此，最多可以计到 2^{16}，即 65536 次。如果单片机的晶体振荡器频率为 6 MHz，则最长的定时时间为

$$t_d=2^{16}\times T_0=2^{16}\times\frac{1}{f}=65536\times 1\ \mu s=65536\ \mu s$$

3.　工作方式 2 和工作方式 3

工作方式 2 和工作方式 3 都是 8 位的定时/计数器工作方式，因此最多可以计到 2^8，即 256 次。如果单片机的晶体振荡器频率为 6 MHz，则最长的定时时间为

$$t_d=2^8\times T_0=2^8\times\frac{1}{f}=256\times 1\ \mu s=256\ \mu s$$

预置值计算：用最大计数值减去所需要的计数次数即可。例如流水线上一个包装要求为 24 盒，要求每到 24 盒就产生一个动作，若用单片机的工作方式 0，则预置值为 8192−24=8168。若假设预置为 N，则定时时间为

$$t_d=(2^{13}-N)\times T_0=(2^{13}-N)\times\frac{1}{f}=(8192-8168)\times 1\ \mu s=24\ \mu s$$

5.4　如何设计定时控制器

1. 定时/计数器初始化程序的编写

（1） 选择定时／计数器的工作模式，即赋值给 TMOD。
（2） 送定时／计数器的初值，即赋值给 TH0、TL0（或 TH1、TL1）。
（3） 开中断，启动定时器中断。
（4） 启动定时／计数器，将 TR0 或 TR1 置 1。

2. 定时方波控制器

例如要用定时/计数器实现在单片机 P1.0 端口输出一个周期为 2 s 的方波，我们采用定时/计数器的工作方式 1，则可以得到最大的定时时间 65536×1=65536 μs，如果我们定时在 0.05 秒，则设置如下所示。

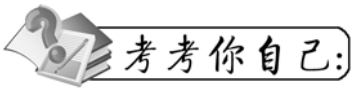
工作方式：TMOD=00000001B=01H

时间初值：65536−50000=15536=3CB0H，所以：TH0=3CH，TL0=B0H

具体程序：

```
        ORG 0000H
        AJMP START
        ORG 30H
START:  MOV P1,#0FFH            ；关所有灯
        MOV TMOD,#00000001B     ；定时/计数器 0 工作于方式 1
        MOV TH0,#3CH
        MOV TL0,#0B0H           ；即数 15536
        SETB TR0                ；定时/计数器 0 开始运行
LOOP:   JBC TF0,NEXT            ；如果 TF0 等于 1，则清 TF0 并转 NEXT 处
        AJMP LOOP               ；否则跳转到 LOOP 处运行
NEXT:   CPL P1.0
        MOV TH0,#3CH
        MOV TL0,#0B0H           ；重置定时/计数器的初值
        AJMP LOOP
        END
```

TF0 为定时/计数器的溢出标志位，当定时器产生溢出后，该位由 0 变 1，所以查询该位就可知定时时间是否已到。该位由 0 变为 1 后，要用软件将标记位清零，以便下一次定时时间到时该位由 0 变 1。



1. 用定时/计数器定时，在 P1.8 引脚上产生周期为 100 ms 的方波。

2. 用定时/计数器 T0 计数，当 P3.4 引脚上脉冲信号满 50 个，使 P1.7 引脚上控制的 LED 灯点亮。（低电平点亮）

项目六　一位数码显示器设计

——数码管静态显示

教学目的

掌握：数码管静态显示接口电路设计方法。
理解：查表指令的应用；静态显示程序设计方法。
了解：数码管的内部结构。

愿你知多点：

在日常生活中，我们经常使用各种显示器，例如交通灯时间显示器、数字温度计显示屏、压力表显示屏等，那么这些显示器是如何制作的呢？在这一章中，我们将通过完成一位数码显示器设计这一任务来学习数码管静态显示器的方法及相关知识。图 6-1 所示为数码管应用实例。

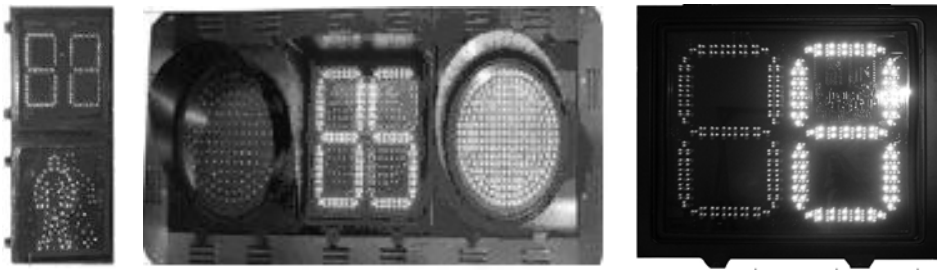


图 6-1　数码管应用实例

6.1　能力培养

本章通过完成一位数码显示器设计这一任务，可以培养以下能力。

1. 能识别数码管的类型和引脚。
2. 能正确使用数码管。
3. 能制作 9 s 计数器。

6.2　任务分析

要完成此项任务，需要掌握以下 3 方面知识。

1. 如何使用数码管。
2. 如何设计数码管与单片机接口电路。
3. 如何设计数码管显示程序。

6.3 如何使用数码管

6.3.1 数码管的内部结构

图 6-2 所示为数码管实物图，数码管引脚排列如图 6-3 所示。由图 6-3 可以看出，数码管共有 10 只引脚，其中 8 只引脚排列为 a、b、c、d、e、f、g、dp，即 8 个笔画，另外两只引脚（即第③脚和第⑧脚）为数码管的公共端，公共端在数码管内部是相互连通的。

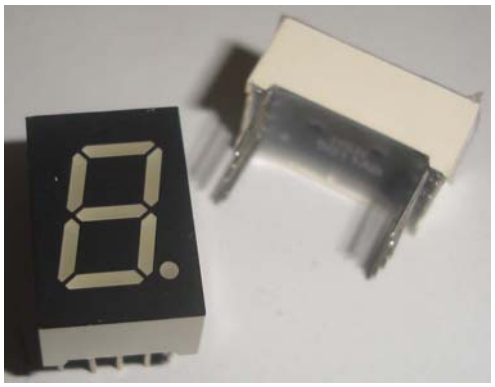


图 6-2 数码管实物

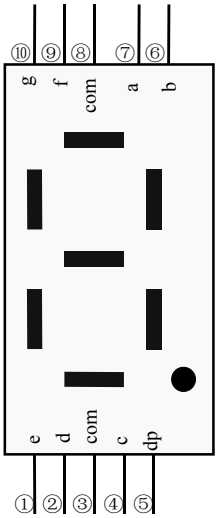


图 6-3 数码管引脚排列

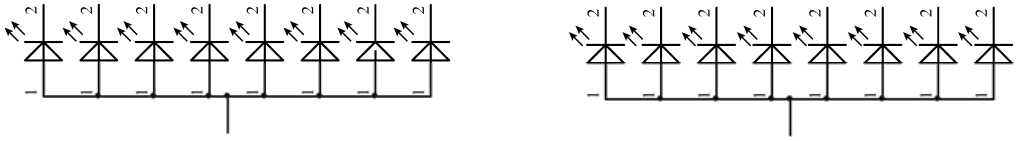
温馨提示：

在图 6-3 中，从数码管的正面看，以左下侧第一脚为起点，按逆时针方向排序，引脚分别为：①—e，②—d，③—com，④—c，⑤—dp，⑥—b，⑦—a，⑧—com，⑨—f，⑩—g。

6.3.2 数码管的类型

图 6-4 所示为数码管类型，数码管根据公共端的连接方式，可以分为共阴数码管和共阳数码管两种。图 6-4（a）所示为共阴数码管，其特点是将 8 只发光二极管的负极接在一起。图 6-4（b）所示为共阳数码管其特点是将 8 只发光二极管的正极接在一起。下面将以

共阳数码管为例，介绍数码管的使用方法。



（a）共阴数码管　　（b）共阳数码管

图 6-4　数码管类型

首先，将共阳数码管的公共端 COM（即发光二极管的正极）接高电平，然后根据需要显示的数字，在相应的笔划端（即发光二极管的负极）接上低电平，点亮所需要的笔划。例如，要显示数字“3”，则在 a、b、c、d、g 端加低电平，那么 a、b、c、d、g 共 5 个笔划点亮，其余端加高电平，不点亮，其他显示数字原理与之相同。8 段 LED 数码管显示字符与笔划编码对应表见表 6-1 所示。

表 6-1　8 段 LED 数码管显示字符与笔划编码对应表

数　字	字　形	共阳数码管									共阴数码管								
		dp	g	f	e	d	c	b	a	编　码	dp	g	f	e	d	c	b	a	编　码
0		1	1	0	0	0	0	0	0	C0H	0	0	1	1	1	1	1	1	3FH
1		1	1	1	1	1	0	0	1	F9H	0	0	0	0	0	1	1	0	06H
2		1	0	1	0	0	1	0	0	A4H	0	1	0	1	1	0	1	1	5BH
3		1	0	1	1	0	0	0	0	B0H	0	1	0	0	1	1	1	1	4FH
4		1	0	0	1	1	0	0	1	99H	0	1	1	0	0	1	1	0	66H
5		1	0	0	1	0	0	1	0	92H	0	1	1	0	1	1	0	1	6DH
6		1	0	0	0	0	0	1	0	82H	0	1	1	1	1	1	0	1	7DH
7		1	1	1	1	1	0	0	0	F8H	0	0	0	0	0	1	1	1	07H
8		1	0	0	0	0	0	0	0	80H	0	1	1	1	1	1	1	1	7FH
9		1	0	0	1	0	0	0	0	90H	0	1	1	0	1	1	1	1	6FH
无显示		1	1	1	1	1	1	1	1	FFH	0	0	0	0	0	0	0	0	00H

6.4　如何设计数码管与单片机接口电路

图 6-5 所示为数码管与单片机接口电路，将单片机 89C51 的 P2.6 端口接到三极管 VT1 的基极，通过控制 VT1 的导通状态来控制共阳数码管公共端电压的高低；单片机 P1 端口的电平经过数据缓冲器 U2 送到数码管笔划端 a～g 和 dp。

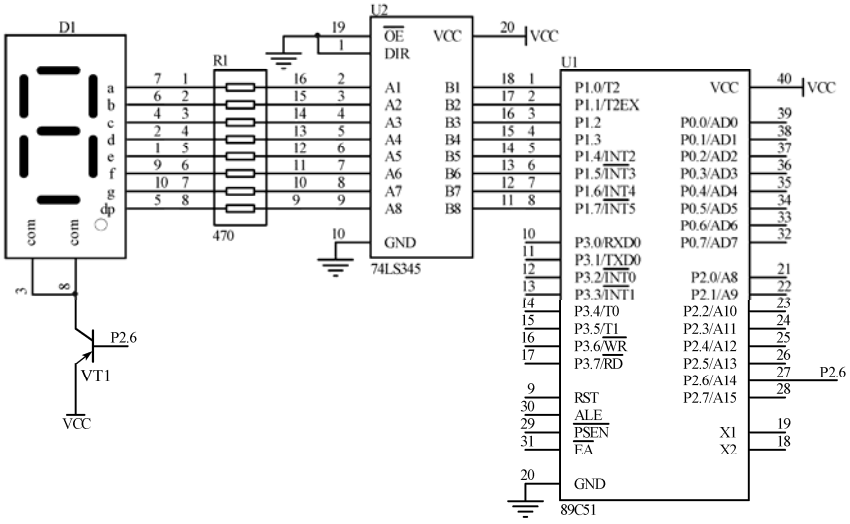


图 6-5　数码管与单片机接口电路

温馨提示：

这种电路中的数码管一直处于显示状态，因此称静态显示。

6.5　如何设计数码管显示程序

要固定显示某一个数字（即日期的个位数字），只要数码管公共端固定保持高电平，同时 P1 端口输出对应字符编码即可。

例如假设日期的个位数为“6”，则显示程序清单如下所示。

```
MAIN:   CLR   P2.6          ；P2.6 端口输出低电平，则共阳数码管公共端为高电平
        MOV   P1，#82H      ；将字符编码 82H 输出到 P1 端口，显示“6”
        SJMP  MAIN          ；循环显示
```

考考你自己：

1. 请简述 8 段 LED 数码管的内部结构。
2. 共阴数码管与共阳数码管有何区别？
3. 软件延时与定时/计数器定时各有什么优缺点。
4. 请设计一个周期在 0～9 s 的计数器。

项目七　学号显示器设计

——数码管动态显示

教学目的

掌握：数码管动态显示接口电路设计方法。
理解：动态显示程序设计方法。
了解：数码管动态显示原理。

愿你知多点：

在上一章中，我们已学习了数码管的基本使用，但在我们的日常生活中，通常要显示的数据不止一位，那么多位显示器是如何制作的呢？在这一章中，我们将通过完成学号显示器设计这一任务来学习数码管动态显示器的制作方法及相关知识。

7.1　能力培养

本章通过完成学号显示器设计这一任务，可以培养以下能力。

1. 能正确连接数码管动态显示电路。
2. 能正确编写数码管动态显示程序。

7.2　任务分析

要完成此项任务，需要掌握以下 3 方面知识。

1. 数码管动态显示原理。
2. 如何设计数码管与单片机动态显示接口电路。
3. 如何设计数码管动态显示程序。

7.3　数码管动态显示原理

在数码管静态显示器中，每位数码管要接 8 个 I/O 端口，要显示多位数据时，需要更多的 I/O 端口，而单片机的 I/O 端口资源有限，为此我们可以采用动态显示解决这一问题。

数码管动态显示其实就是多个数码管交替显示，它利用了人眼的视觉暂留特性，让人

看到好像有多个数码管同时显示。在编程时需要输出字段和字位信号，字位信号用于选中其中一位数码管，然后输出字段码，让该数码管显示所需要的内容，延时一段时间后，再选中另一个数码管，并输出对应的字段码，让数码管高速交替显示。例如显示数字“24”时，先输出字位信号，选中十位数码管，输出“2”的字段码，延时一段时间后选中个位数码管，输出“4”的字段码。再以一定速度循环执行上述过程，就可以显示出“24”，由于交替的速度非常快，人眼看到的就是连续的“24”。在动态显示程序中，各个位的延时时间长短是非常重要的，如果延时时间长，则会出现闪烁现象；如果延时时间太短，则会出现显示暗且有重影的现象。

7.4 如何设计数码管与单片机动态显示接口电路

数码管与单片机动态接口电路如图 7-1 所示。其中 RP 为排组，DS0、DS1 为两个共阴数码管，VT1、VT2 为驱动三极管。

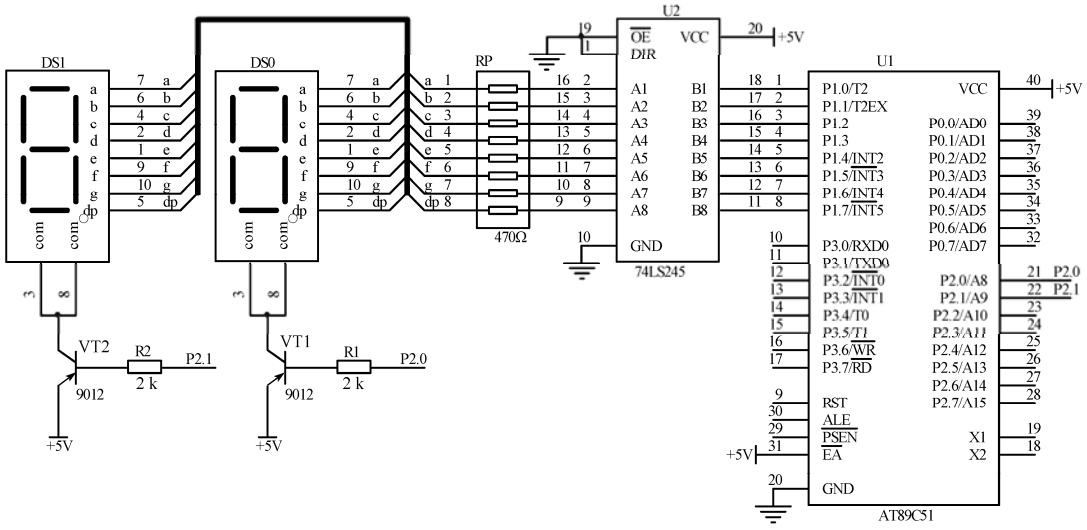


图 7-1 数码管与单片机动态接口电路

单片机的 P1 端口输出的字段码经 74LS245 驱动、RP 限流后分别加到数码管 DS0 和 DS1 的 a～dp 端，显示对应的字形；单片机的 P2.0 和 P2.1 端口输出字位信号，控制对应数码管点亮。

7.5 如何设计数码管动态显示程序

7.5.1 学号显示器程序流程图

学号显示器程序流程图如图 7-2 所示。

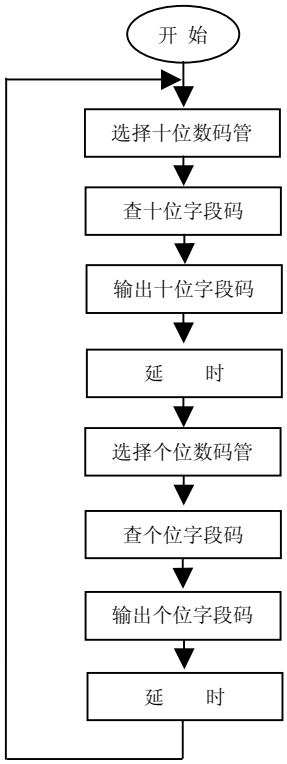


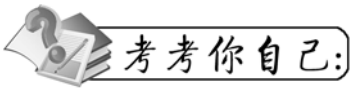
图 7-2　学号显示器程序流程图

7.5.2　学号显示器程序

若要显示的学号为 24，则程序清单如下所示。

```
ORG 0000H
MIAN:   SETB P2.0          ；关闭个位显示器
        CLR P2.1           ；点亮十位显示器
        MOV A,#02H         ；学号的十位数“2”送入 A
        MOV DPTR,#TABLE ；DPTR 指向表首址
        MOVC A,@A+DPTR  ；查表，取出“2”对应的字形码
        MOV P1,A           ；送 P1 端口显示
        CALL DELAY         ；调用延时子程序
        SETB P2.1          ；关闭十位显示器
        CLR P2.0           ；点亮个位显示器
        MOV A,#02H         ；学号的个位数“4”送入 A
        MOV DPTR,#TABLE ；DPTR 指向表首址
        MOVC A,@A+DPTR  ；查表，取出“4”对应的字形码
        MOV P1,A           ；送 P1 端口显示
```

```
        CALL DELAY      ；调用延时子程序
        LJMP START
DELAY:  MOV R1,#0FH     ；延时子程序
LOOP1:  MOV R2#0FFH
LOOP2:  DJNZ R2,LOOP2
        DJNZ R1,LOOP1
        RET
```



1. 在学号显示器电路中，VT1 和 VT2 有何作用？
2. 请设计一个秒表。
3. 动态显示程序中，若延时子程序的延时时间太长，有何不良后果。
4. 在学号显示器程序中，若漏写关闭显示器程序，能否正常显示学号？

项目八　一位汉字显示屏的设计

——LED 点阵显示屏

教学目的

掌握：LED 点阵显示接口电路设计方法。
理解：LED 点阵显示器程序设计方法。
了解：LED 显示屏的结构。

愿你知多点：

在日常生活中，我们经常使用各种复杂显示器，不但能显示数字、字符（前面章节所介绍），还要能显示各种复杂的图案与汉字，例如各种广告屏。那么这些显示器是如何设计制作的呢？在这一章中，我们将通过完成一位汉字显示屏的设计这一任务来学习 LED 点阵显示屏的制作方法及相关知识。图 8-1 所示为 LED 点阵显示屏应用实例。

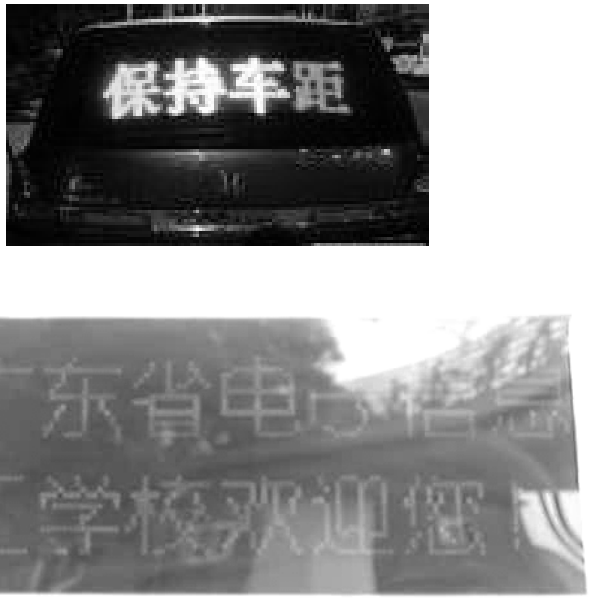


图 8-1　LED 点阵显示屏应用实例

8.1 能力培养

本章通过完成一位汉字显示屏的设计这一任务，可以培养以下能力。

1. 能正确使用 LED 点阵显示屏。
2. 能正确设计 LED 点阵显示接口电路。
3. 能编写 LED 点阵显示器程序。

8.2 任务分析

要完成此项任务，需要掌握以下 3 方面知识。

1. 如何使用 LED 点阵显示屏。
2. 如何设计汉字点阵显示电路。
3. 如何编写汉字点阵显示程序。

8.3 如何使用 LED 点阵显示屏

8.3.1 如何显示汉字

单片机中的汉字显示器一般采用点阵方式显示。为了在 8×8 LED 发光二极管点阵显示屏上显示汉字，首先要把汉字表示成为 8×8 像素点图，图 8-2 所示为汉字“出”的 8×8 像素点图。

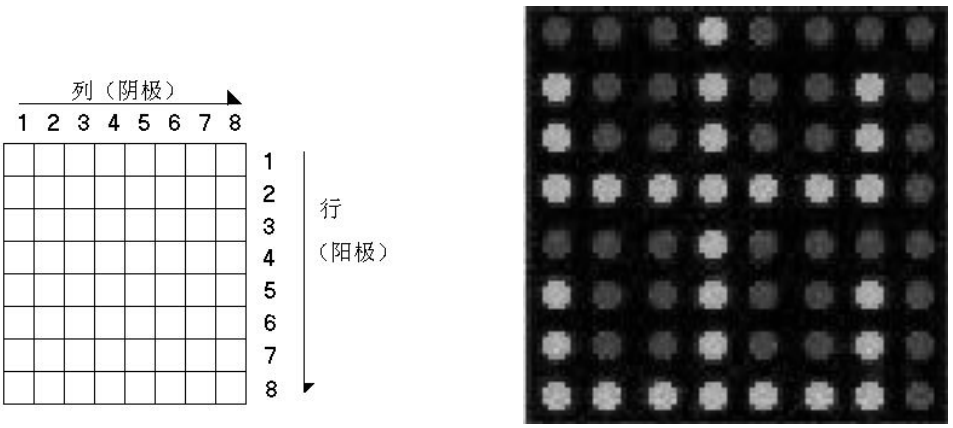


图 8-2 汉字“出”的 8×8 汉字像素点图

如果用“1”表示点亮像素，“0”表示暗像素，则一个汉字可以用 8 个字节表示像素，称为该汉字的字模。例如“出”的字模为 0×10，0×92，0×92，0×FE，0×10，0×92，0×92，0×FE。

这样，要在 LED 点阵显示器上显示汉字，只要按该汉字的字模点亮相应的像素点即可。

8.3.2　如何使用 LED 点阵显示屏

8×8 LED 点阵显示屏实物图如图 8-3 所示，其内部结构如图 8-4 所示。

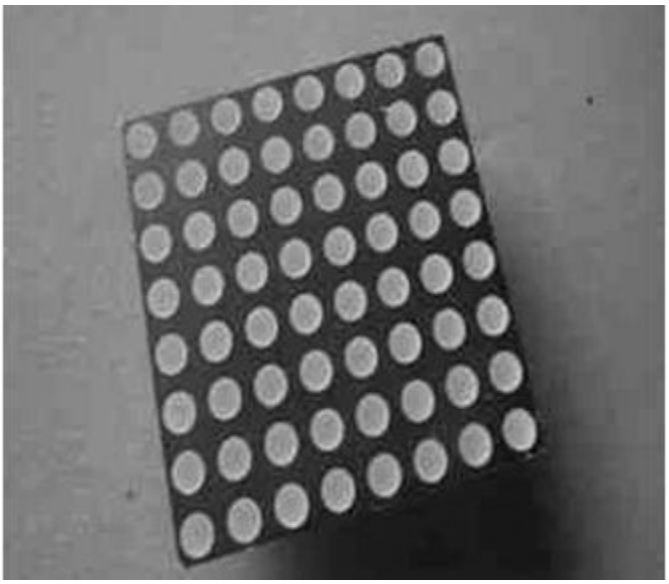


图 8-3　8×8 LED 点阵显示屏实物图

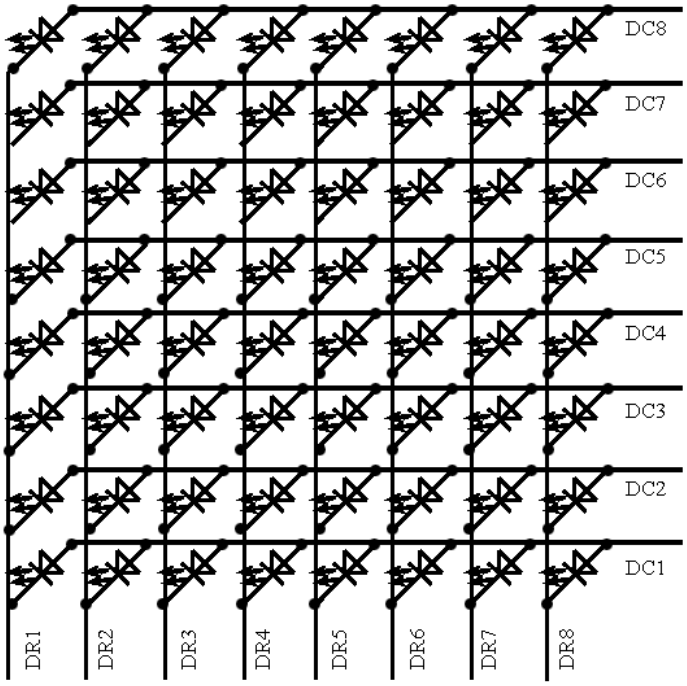


图 8-4　8×8 LED 点阵显示屏内部结构

从图 8-4 中可以看出，LED 点阵显示屏内部是由发光二极管按行与列排列而成。8×8 屏有 64 个 LED 二极管（点）。每 8 个二极管的共阳极为一行，共阴极为一列，所以有 8 行和 8 列。有 16 只引脚，其中 8 只引脚为行线，分别为 DC1(0)、DC2(1)、DC3(2)、DC4(3)、DC5(4)、DC6(5)、DC7(6)和 DC8(7)；8 只引脚为列线，分别为 DR1(A)、DR2(B)、DR3(C)、DR4(D)、DR5(E)、DR6(F)、DR7(G)、DR8(H)，图 8-5 所示为 8×8 LED 点阵显示屏外观及引脚图。

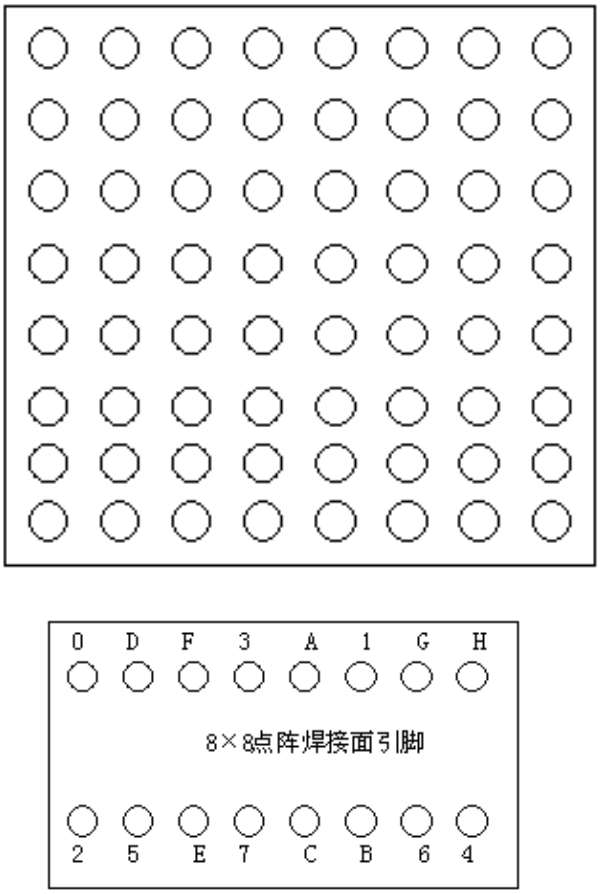


图 8-5　8×8 LED 点阵显示屏外观及引脚图

8.4　如何设计汉字点阵显示电路

汉字点阵显示电路原理图如图 8-6 所示。图中单片机 89C51 的 P1 端口控制行扫描（行 Y0～Y7），用 74HC164 控制字模输出（列 X0～X7），D3（8550×8）为行和列的驱动电路，为 LED 发光二极管提供驱动电流。

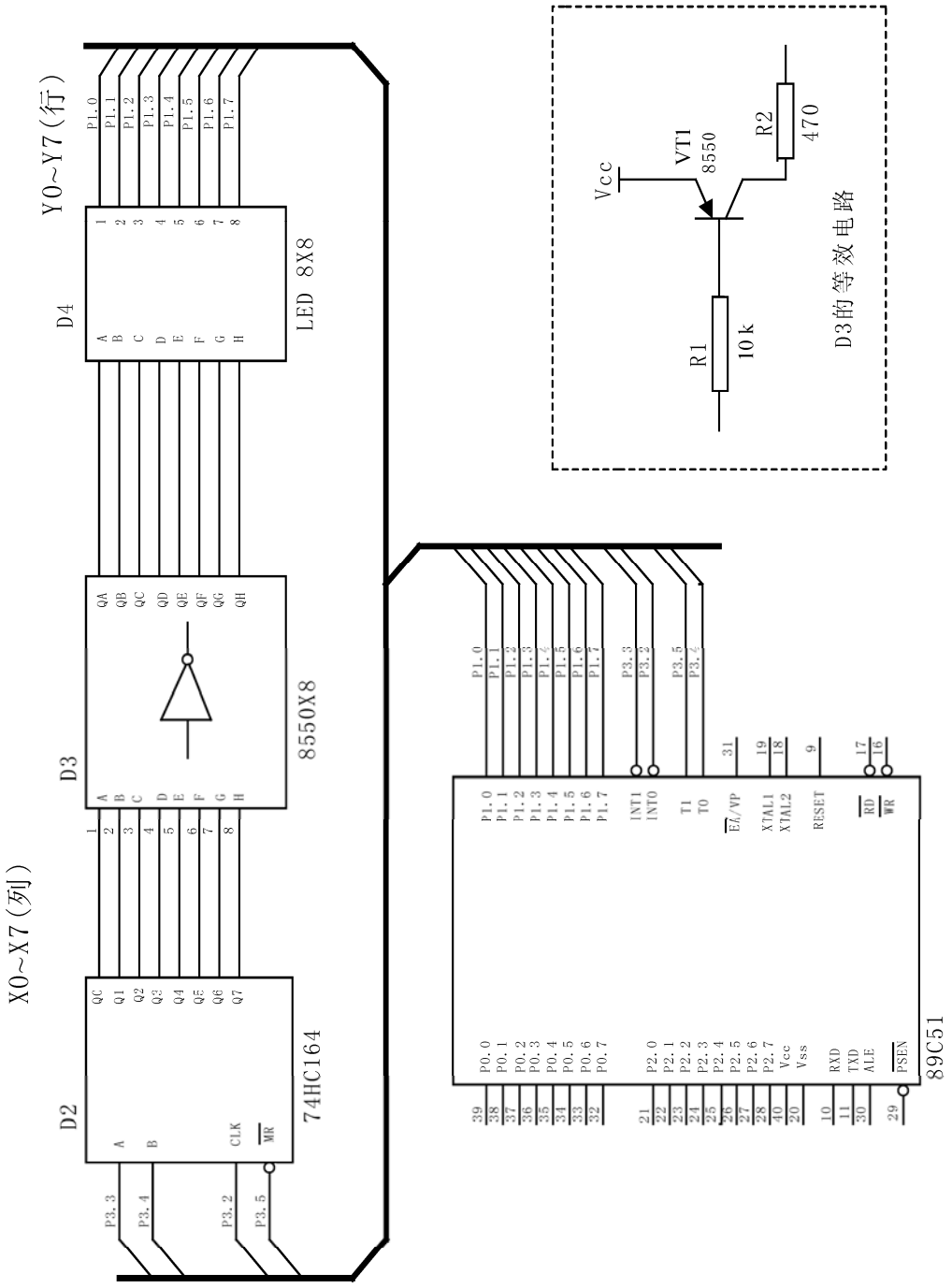


图 8-6　汉字点阵显示电路原理图

8.5 如何编写汉字点阵显示程序

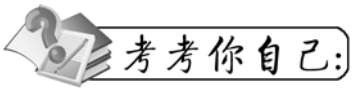
LED 点阵显示屏的扫描可以分为点扫描、行扫描和列扫描 3 种。

根据视觉暂留要求，点扫描的频率为 16×64=1024 Hz，即周期小于 1 ms。行扫描和列扫描频率必须大于 16×8=128 Hz，即周期小于 7.8 ms。

下面以 8×8 点阵列扫描显示汉字“出”为例，介绍汉字的显示方法。

```
    ORG 0000H
START:
    MOV SP,#70H          ；设堆栈指针
    MOV 50H,#10H         ；设置发送的列数据（X0～X7），“出”的字模数据
    MOV 51H,#92H
    MOV 52H,#92H
    MOV 53H,#0FEH
    MOV 54H,#10H
    MOV 55H,#92H
    MOV 56H,#92H
    MOV 57H,#0FEH
    CLR CLEAR            ；初始化 I/O 端口
    SETB CLK
    SETB DINA
    SETB DINB
    SETB CLEAR
    MOV R6,#08H          ；设置扫描次数
    MOV DPTR,DB          ；读取扫描端口数据
    MOV R1,#50H          ；指定列数据指针
flashcy:  MOV A,@R1      ；读取列数据
    MOV R0,A
    INC R1               ；列数据指针加 1
    CLR A
    MOVC A,@A+DPTR
    MOV P1,#0FFH
    LCALL SENDTO
    MOV P1,A
    INC DPTR
    LCALL DELAY
    DJNZ R6,flashcy
```

```
        SJMP START
    Sendto:                 ；发送数据子程序
        PUSH ACC
        CLR CLK
        MOV R7,#08H
        MOV A,R0
        CLR C
    SENDCY: RRC A
        MOV DINA,C
        SETB CLK
        CLR CLK
        DJNZ R7,SENDCY
        POP ACC
        RET
    DELAY: MOV R7,#01       ；延时子程序
        DJNZ R3,$
        DJNZ R4,DELAY1
        DJNZ R7,DELAY0
        RET
        DB 0FEH,0FDH,0FBH,0F7H,0EFH,0DFH,0BFH,07FH  ；Y0～Y7（行扫描）
```



1. LED 点阵显示器的内部结构有何特点？

2. 请设计一个 16×16 汉字显示器，其中硬件电路如图 8-7 所示，LED 显示屏为 16×16 点阵显示器，要求循环显示“我爱单片机”，请编写相关程序。

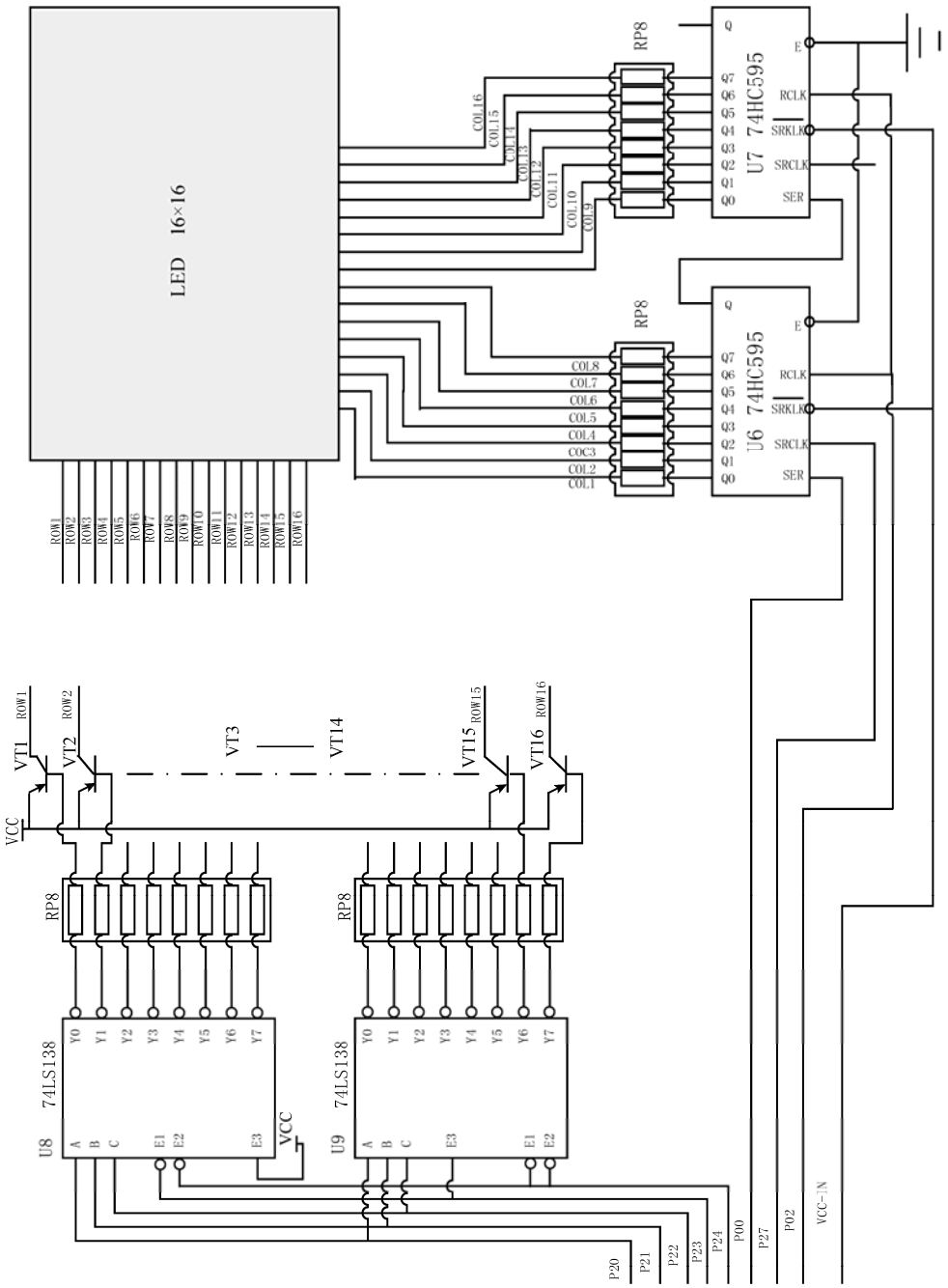


图 8-7　16×16 汉字显示器的硬件电路

项目九　模拟数字式温度计

——A/D 转换及其与单片机接口技术

教学目的

掌握：ADC0809 与单片机接口电路设计方法。
理解：A/D 转换原理。
了解：A/D 转换程序设计方法。

愿你知多点：

在单片机测控系统中，通常需要采集一些连续变化的物理信号即模拟信号，如温度、速度、电压、电流、压力等，但单片机只能加工和处理数字信号，因此在单片机测控系统中处理模拟信号时，需要进行模/数转换，即 A/D 转换。

9.1　能力培养

本章通过完成模拟数字温度计这一任务，可以培养以下能力。

1. 能正确使用 ADC0809。
2. 能正确设计 A/D 转换电路。
3. 能编写 A/D 转换程序。

9.2　任务分析

要完成此项任务，需要掌握以下 4 方面知识。

1. A/D 转换的基础知识。
2. 如何使用 A/D 转换器。
3. 如何设计 A/D 转换器与单片机接口电路。
4. 如何设计 A/D 转换器与单片机接口程序。

9.3　A/D 转换的基础知识

9.3.1　A/D 转换器原理

将模拟信号转换成数字信号的器件称为 A/D 转换器。随着大规模集成电路技术的迅速

发展，A/D 转换器的新品不断推出。A/D 转换器按工作原理可分为逐次逼近式、双积分式、计数比较式、V/F 转换式和并行式 5 种。

在这些转换器中，计数比较式 A/D 转换器比较简单，但转换速度较慢，基本被淘汰。双积分式 A/D 转换器精度高，一般用于数据采集以及对精度要求比较高的场合，但其转换速度慢。V/F 转换式 A/D 转换器主要用于远距离串行传送。并行式 A/D 转换器比较复杂，且成本高，只用在一些对转换速度要求较高的场合。因为逐次逼近式 A/D 转换器既兼顾了转换速度，又具有一定的转换精度，所以目前大多使用逐次逼近式 A/D 转换器。下面将以逐次逼近式 A/D 转换器和双积分式 A/D 转换器为例，介绍 A/D 转换原理。

1. 逐次逼近式 A/D 转换器

逐次逼近式 A/D 转换是用一个计量单位将连续量整量化（简称量化），即用计量单位与连续量进行比较，把连续量转换为计量单位的整数倍，略去小于计量单位的连续量部分。这样所得到的整数量即为数字量。显然，计量单位越小，量化的误差也越小。N 位逐次逼近式 A/D 转换器方框图如图 9-1 所示。

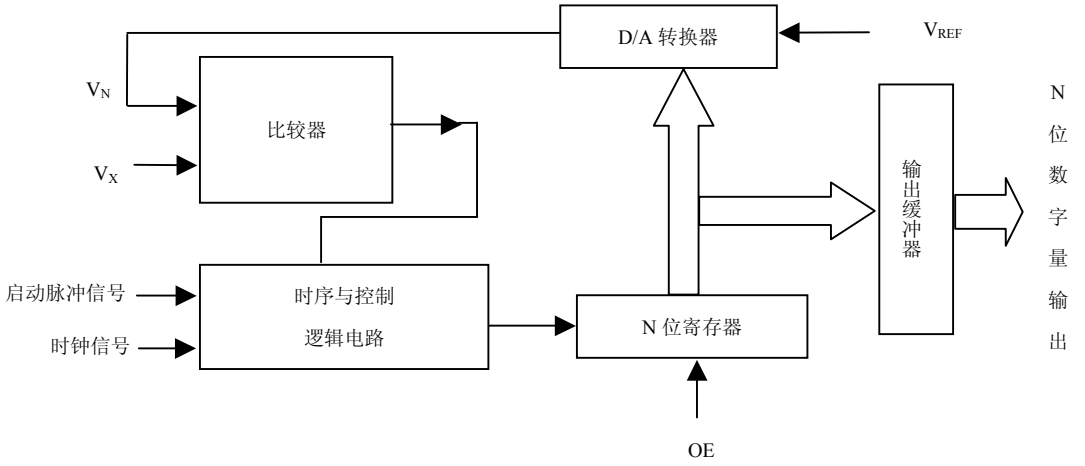


图 9-1　N 位逐次逼近式 A/D 转换器方框图

逐次逼近式 A/D 转换器由 N 位寄存器、D/A 转换器、比较器及时序与控制逻辑电路等部分组成。N 位寄存器用来存放 N 位二制数码。

转换开始时，将 N 位寄存器清零，这时 D/A 转换器输出的电压也为零。当 A/D 转换器接到启动脉冲信号后（即有模拟信号 V_X 输入后），在时钟信号的作用下，启动信号通过控制逻辑电路启动 A/D 转换器。首先，将 N 位寄存器最高位（D_{N-1}）置“1”，其余位清零，N 位寄存器的内容经 D/A 转换后得到整个量程一半的模拟电压 V_N，再将 V_N 与输入电压 V_X 比较。若 $V_N < V_X$，则保留 $D_{N-1}=1$；若 $V_N > V_X$，将 D_{N-1} 位清零。然后，控制逻辑使寄存器下一位（D_{N-2}）置“1”，再与上次的结果一起经 D/A 转换器后与 V_X 比较，如此继续下去，直到判断出 D_0 位比较完成为止，此时时序与控制逻辑电路发出转换结束标志 EOC。这样经过 N 次比较后，N 位寄存器的内容就是转换后的数字量，在输出允许信号 OE 有效的条件下，此值经输出缓冲器输出。整个转换过程就是一个逐次比较逼近过程。

逐次比较逼近 A/D 转换器只需要比较 N 次就可以转换成对应的数字量，转换速度比较快，因此目前大多数 A/D 转换器都采用这种方法，常用的逐次逼近式 A/D 转换器的型号有 ADC0809、AD574A 等。

2. 双积分式 A/D 转换器

双积分式 A/D 转换采用间接测量原理，即将被测电压值 V_X 转换成时间常数，通过测量时间常数得到未知电压值，双积分式 A/D 转换器方框图如图 9-2 所示。

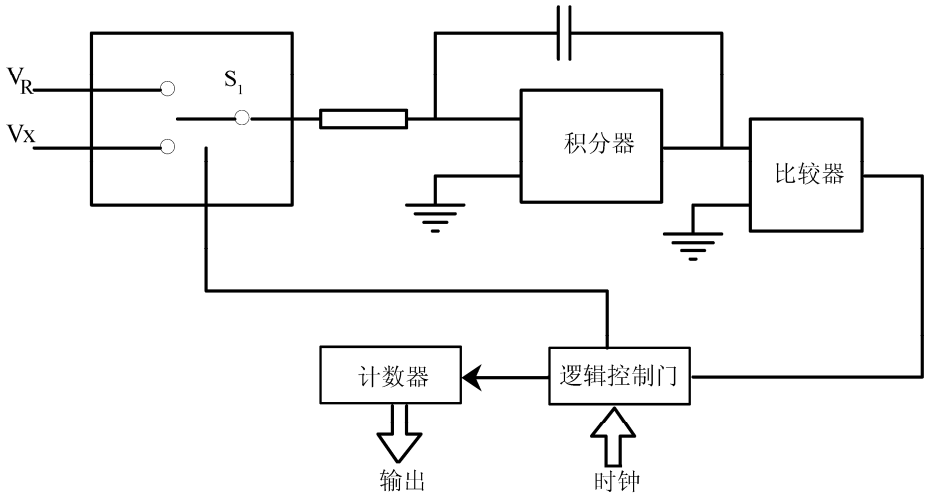


图 9-2　双积分式 A/D 转换器方框图

双积分式 A/D 转换器由电子开关、积分器、比较器、计数器、逻辑控制门等部件组成。其中积分器是双积分式 A/D 转换器的核心部分，其输入端所接的开关 S_1 由逻辑控制门控制。比较器用来判断积分器输出电压过零的时间。

双积分式 A/D 转换器原理如图 9-3 所示，双积分本质就是进行一次 A/D 转换需要两次积分，具体过程如下所示。

第一次积分，即正向积分。转换时，逻辑控制门通过电子开关把被测电压 V_X 加到积分器的输入端，积分器从零开始，在固定时间 T_0 内对 V_X 进行正向积分（称定时积分），积分输出终值与 V_X 成正比。

第二次积分，即反向积分。在完成正向积分后，逻辑控制门将电子开关切换到极性 与 V_X 相反的基准电压 V_R 上，进行反向积分，由于基准电压 V_R 恒定，所以积分输出将按 T_0 期间积分的值以恒定的斜率下降，当比较器检测到积分输出过零时，积分器停止工作。反向积分时间 T_1 与定值积分的初值（即定时积分的终值）成比例关系，故可以通过测量反向积分时间 T_1 计算出 V_X，即 $V_X=\frac{T_1}{T_0}V_R$。反向积分时间 T_1 由计数得到。

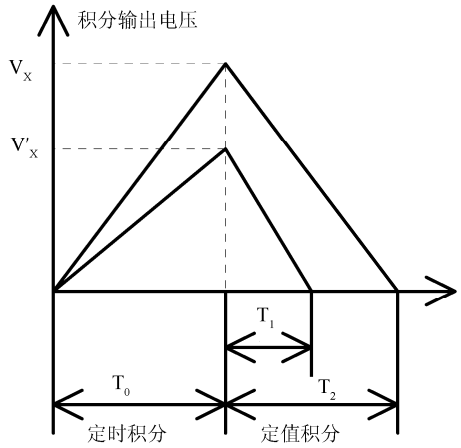


图 9-3　双积分式 A/D 转换器原理

图 9-3 显示出了两种不同输入电压（V_X>V'_X）的积分情况，显然 V'_X 值小，在 T_0 定时积分期间积分器输出终值也就小，而下降斜率相同，故反向积分时间 T'_1也就小。

由于积分可抑制模拟量输入信号噪音，因此双积分式 A/D 转换器抗干扰能力强，且精度高，但采用正向与反向积分，需要两次积分，这样会使工作速度较慢。转换时间与输入信号大小有关，一般为几十毫秒。

9.3.2　性能指标

1.　分辨率

分辨率用来表示 A/D 转换器对输入信号微小变化的敏感程度，通常用 A/D 转换器输出数字量的位数来表示，如 8 位、10 位、12 位分辨率等，8 位 A/D 转换器的分辨率就是 8 位，或者说分辨率为满刻度的 $1/2^8$=1/256。分辨率越高，对输入信号微小变化的反应越灵敏。

2.　量程

即所能转换的电压范围，如 5 V、10 V。

3.　转换精度

A/D 转换器的精度是指与数字输出量所对应的模拟输入量的实际值与理论值之间的差值。A/D 转换器中与每个数字输出量对应的模拟输入量并非是一个单一的数值，而是一个范围值Δ，其中Δ的大小理论上取决于电路的分辨率。定义Δ为数字输出量的最小有效位 LSB。但在外界环境的影响下，与每一个数字输出量对应的模拟输入量实际范围往往偏离理论值Δ。

4.　转换时间

转换时间是指 A/D 转换器完成一次转换所需要的时间，转换时间通常以 A/D 转换器的

时钟周期为单位进行定义。

9.4　如何使用 A/D 转换器

ADC0809 是一个典型的 8 位 8 通道逐次逼近式 A/D 转换器，可实现 8 路模拟信号的分时采集，其转换时间为 100 μs 左右，采用双列直插式封装，共有 28 只引脚。ADC0809 引脚排列如图 9-4 所示。

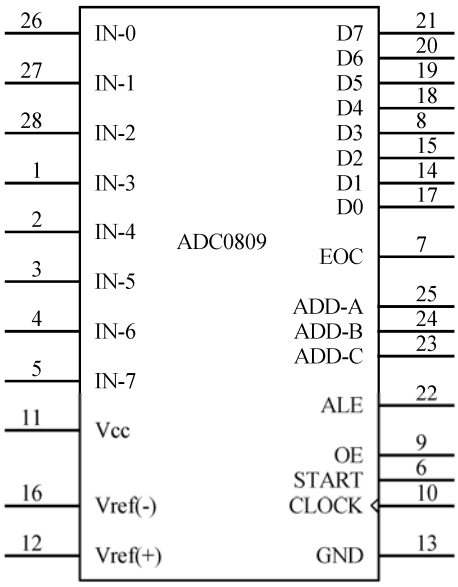


图 9-4　ADC0809 引脚排列

IN-7～IN-0：8 路模拟量输入端。ADC0809 对模拟输入量的要求主要有单极性，电压范围 0～5 V，若输入的信号过小还需进行放大。

ADD-A、ADD-B、ADD-C：3 根通道端口选择线。ADD-A 为低位地址，ADD-C 为高位地址，用于选择 8 路模拟输入量，其地址状态与所选模拟输入量的对应关系见表 9-1 所列。

表 9-1　ADD-A、ADD-B 和 ADD-C 与所选模拟输入量

ADD-C	ADD-B	ADD-A	所选模拟输入量
0	0	0	IN-0
0	0	1	IN-1
0	1	0	IN-2
0	1	1	IN-3
1	0	0	IN-4
1	0	1	IN-5
1	1	0	IN-6
1	1	1	IN-7

D0～D7：8 位数字量输出端。

ALE：地址锁存允许信号。ALE 为高电平时，将 ADD-A、ADD-B、ADD-C 的地址状态送入地址锁存器中。

START：A/D 转换启动信号。当 START 为高电平时，启动 A/D 转换；在 A/D 转换期间，START 应保持低电平。

OE：输出允许信号，用于控制三态输出锁存器向单片机输出转换得到的数据，高电平有效。当 OE＝0 时，输出数据线呈高电阻状态；当 OE＝1 时，A/D 转换数据输出到数据线 D0～D7。

CLOCK：时钟信号。ADC0809 的内部没有时钟电路，所需时钟信号由外部提供，因此 ADC0809 设有时钟信号引脚，其时钟信号的频率为 500 kHz。

EOC：转换结束信号。A/D 转换期间 EOC 为低电平；A/D 转换结束时 EOC 为高电平。

Vcc：+5 V 电源。

GND：地端。

Vref(+)、Vref(−)：基准电压输入端。基准电压用于和输入模拟量进行比较，作为逐次逼近的基准电压，其典型值为+5 V（Vref(+)=+5 V，Vref(−)=0 V）。

9.5　如何设计 A/D 转换器与单片机接口电路

ADC0809 与单片机的接口电路如图 9-5 所示。图中单片机 D1（89C51）的地址总线 A_0、A_1、A_2（即 P00、P01、P02）通过锁存器 U4（74LS373）分别接 ADC0809 的 ADD-A、ADD-B、ADD-C，通过 ADD-A、ADD-B、ADD-C 三只引脚来选择模拟输入量。ADC0809 的 8 根输出数据线 D0～D7 直接与单片机的 P00～P07 相连，并由输出允许信号（OE）控制。

地址锁存允许信号（ALE）和 A/D 转换启动信号（START）由软件产生（执行 MOVX @DPTR，A 指令），输出允许信号（OE）也由软件产生（执行 MOVX　A,@DPTR）。单片机的 ALE 信号经二分频后作为 ADC0809 的 CLOCK 信号。转换结束信号（EOC）经反向后送到单片机的 INT1 引脚，单片机在中断服务子程序中读取 A/D 转换结果，并将结果送 P1 端口显示。

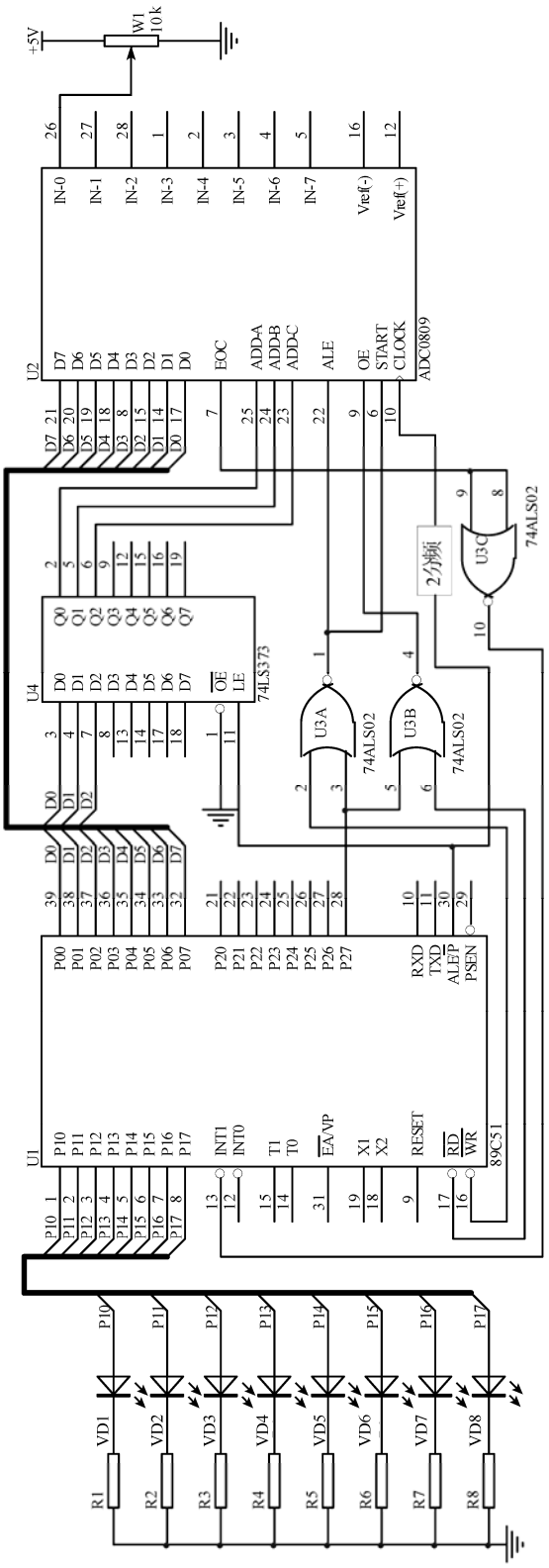
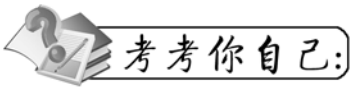


图 9-5　ADC0809 与单片机的接口电路

9.6 如何设计 A/D 转换器与单片机接口程序

单片机执行以下程序，将温度采集显示。

```
        ORG 0000H
        LJMP MAIN           ；转入主程序
        ORG 0013H
        LJMP INT1           ；转入中断服务子程序
MAIN:   SETB EA             ；开总中断
        SETB EX1            ；允许 INT1 中断
        SETB IT1            ；定义中断 1 为下降沿触发
        MOV DPTR，#7FF8H    ；指向 ADC0809 通道 0 地址
        MOVX @DPTR，A       ；启动 A/D 转换
HERE:   SJMP HERE           ；等待 A/D 转换结束
INT1:   MOVX A，@DPTR       ；读取 A/D 转换结果
        MOV P1，A           ；转换结果送 P1 端口显示
        RETI
        END
```



1. A/D 转换器有何作用？为什么要进行 A/D 转换？
2. 简述 A/D 转换原理。
3. A/D 转换器的主要性能指标哪些？并简述其含义。

项目十　锯齿波信号发生器

——D/A 转换及其与单片机接口技术

教学目的

掌握：DAC0832 与单片机接口电路设计方法。
理解：D/A 转换原理。
了解：D/A 转换程序设计方法。

愿你知多点：

在自动控制领域中，通常使用 D/A 转换器把单片机处理的数字量转换成模拟量去控制执行机构。本章将通过完成锯齿波信号发生器这一任务来学习 D/A 转换及其与单片机接口技术的相关知识。

10.1　能力培养

本章通过完成锯齿波信号发生器这一任务，可以培养以下能力。

1. 能正确使用 D/A 转换器 DAC0832；
2. 能正确使用运算放大器 LM324；
3. 能制作 D/A 转换器。

10.2　任务分析

要完成此项任务，需要掌握以下 4 方面知识。

1. D/A 转换的基本知识。
2. 如何使用 D/A 转换器。
3. 如何设计 D/A 转换器与单片机接口电路。
4. 如何设计 D/A 转换器与单片机接口程序。

10.3　D/A 转换的基本知识

将数字量转换成模拟量的器件称为 D/A 转换器，常用的 D/A 转换器有权电阻 D/A 转换器和 T 型电阻网络 D/A 转换器两种，下面以权电阻 D/A 转换器为例对 D/A 转换器进行介绍。

10.3.1 D/A 转换器原理

权电阻 D/A 转换器如图 10-1 所示，它实质上是一只反向求和放大器，主要由权电阻器、位切换开关、反馈电阻器和运算放大器组成。

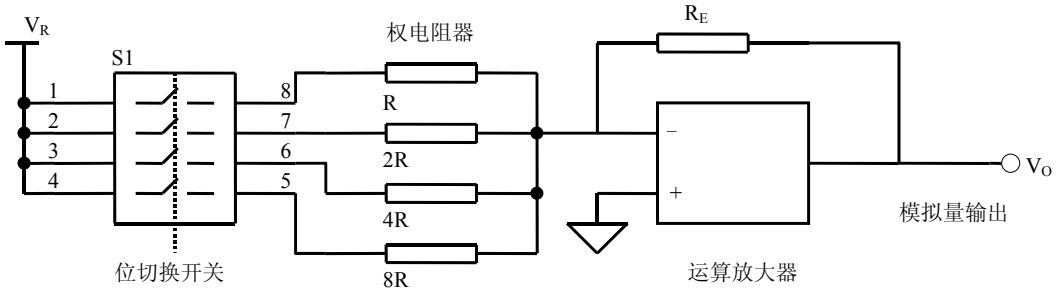


图 10-1 权电阻 D/A 转换器

权电阻器的电阻值按 8∶4∶2∶1 的比例配置（即按二进制权值配制），V_R 为基准电压，这样流过各个权电阻器的电流值分别为 $\frac{V_R}{8R}$、$\frac{V_R}{4R}$、$\frac{V_R}{2R}$、$\frac{V_R}{R}$，各个权电阻器电流的通断由位切换开关控制，这些电流值符合二进制位关系。经运算放大器反向求和，其输出的模拟量与输入的二进制数据 d_3、d_2、d_1、d_0 成比例，等式表示为

$$V_O=-\left(\frac{d_0}{8R}+\frac{d_1}{4R}+\frac{d_2}{2R}+\frac{d_3}{R}\right)R_F \cdot V_R$$

其中，d_3～d_0 是输入二进制的位，取值为 0 时表示位切换开关断开，该位无电流输入；取值为 1 时，表示切换开关合上，该位有电流输入。

选用不同的权电阻网络，就可以得到不同编码的 D/A 转换器。

10.3.2 性能指标

1. 分辨率

分辨率是指满量程信号能分成的步数和阶梯的尺寸，通常用数字量的位数表示，一般为 8 位、10 位、12 位、16 位等。对于 *n* 位的 D/A 转换器，其分辨率为满刻度的 $\frac{1}{2^n}$，例如 10 位 D/A 转换器的分辨率就是 10 位，或者说它可以对满量程 $\frac{1}{2^{10}}=\frac{1}{1024}$ 的增量做出反应。

2. 线性度

线性度也称非线性误差，是实际转换特性曲线与理想转换特性曲线之间的最大偏差，通常用满量程的百分率或最低位（LSB）的分数来表示，如±1%是指实际输出值与理论值的偏差在满刻度的±1%以内。

3. 精度

精度分为绝对精度和相对精度，它是由于非线性、零点刻度、满量程刻度及温漂等因素引起的误差。精度表示 D/A 转换器实际输出值与其理论值的误差。

4. 建立时间

建立时间是指输入数字量发生满刻度变化时，输出模拟信号达到满刻度值的$\pm\frac{1}{2}$ LSB 所需时间，建立时间一般为几微秒到几毫秒。

10.4　如何使用 D/A 转换器

常用到的典型 D/A 转换器型号为 DAC0832，其内部结构方框图如图 10-2 所示，它由输入寄存器、DAC 寄存器和 D/A 转换器三大部分组成。

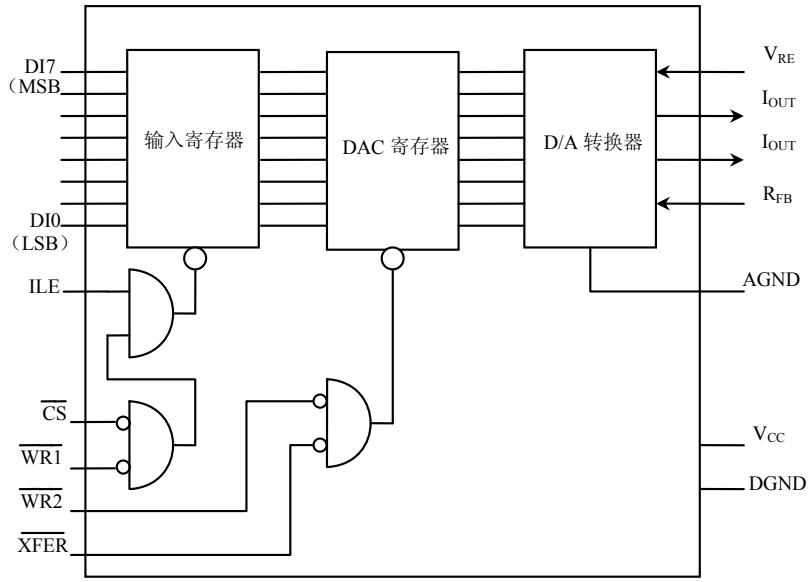


图 10-2　DAC0832 内部结构方框图

DAC0832 是采用 CMOS 工艺制成的单片直流输出型 8 位 D/A 转换器，共 20 只引脚，且采用单电源供电，采用双列直插式封装，引脚排列如图 10-3 所示。电压值范围在+5～+15 V 均可正常工作，基准电压值范围为−10～+10 V，电流建立时间为 1 μs，功耗为 20 mW。

各引脚说明如下所示。

DI0～DI7：数字信号输入端。

$\overline{CS}$：片选信号输入端，低电平有效。

ILE：数据锁存允许控制信号输入端，高电平有效。

$\overline{WR1}$：输入寄存器的写选通信号端，低电平有效。该信号端的信号与 ILE 信号共同控

制输入寄存器的工作方式：当$\overline{CS}$=0，ILE=1，$\overline{WR1}$=0 时，输入寄存器工作在直通方式；当$\overline{CS}$=0，ILE=1，$\overline{WR1}$=1 时，输入寄存器工作在锁存方式。

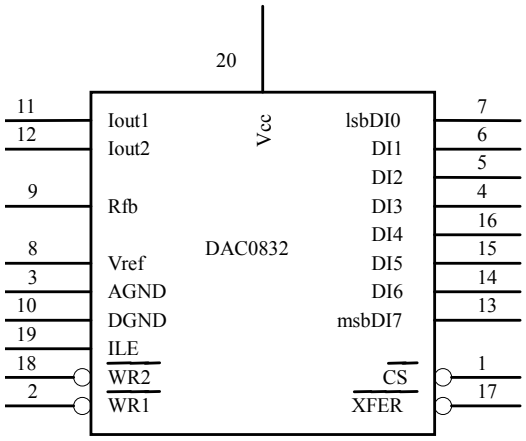


图 10-3　DAC0832 引脚排列

$\overline{XFER}$：数据传送控制信号（输入）端，低电平有效。

$\overline{WR2}$：DAC 寄存器写选通输入端，低电平有效。该信号与$\overline{XFER}$信号共同控制 DAC 寄存器的工作方式，当$\overline{WR2}$=0，$\overline{XFER}$=0 时，DAC 寄存器工作在直通方式；当$\overline{WR2}$=1，$\overline{XFER}$=0 时，DAC 寄存器工作在锁存方式。

Iout1：电流输出 1。当数据全为 0 时输出电流值最小，全为 1 时输出电流值最大。

Iout2：电流输出 2。

注：Iout1+Iout2=常数。

Rfb：反馈端。即 DAC0832 内部反馈电阻器引脚。

Vref：基准电压输入端，输入电压范围为−10 V～+10 V。

Vcc：电源，+5 V～15 V。

DGND：数字地。

AGND：模拟地。

10.5　如何设计 D/A 转换器与单片机接口电路

DAC0832 与单片机的接口电路如图 10-4 所示。

图中 DAC0832 接成单缓冲工作方式。片选信号端$\overline{CS}$和数据传输控制信号端$\overline{XFER}$都接到地址线 A15（即 P27），输入寄存器和 DAC 寄存器地址都可以选为 7FFFFH，写选通信号输入端$\overline{WR1}$、$\overline{WR2}$都与单片机的写信号端$\overline{WR}$连接，单片机对 DAC0832 执行一次写操作，就可以把一个数据直接写入 DAC 寄存器。ILE 接+5 V 电压，Iout2 接地，Iout1 输出电流经运算放大器 U3A（LM324）输出一个单极性电压（范围为−5 V～0 V）。

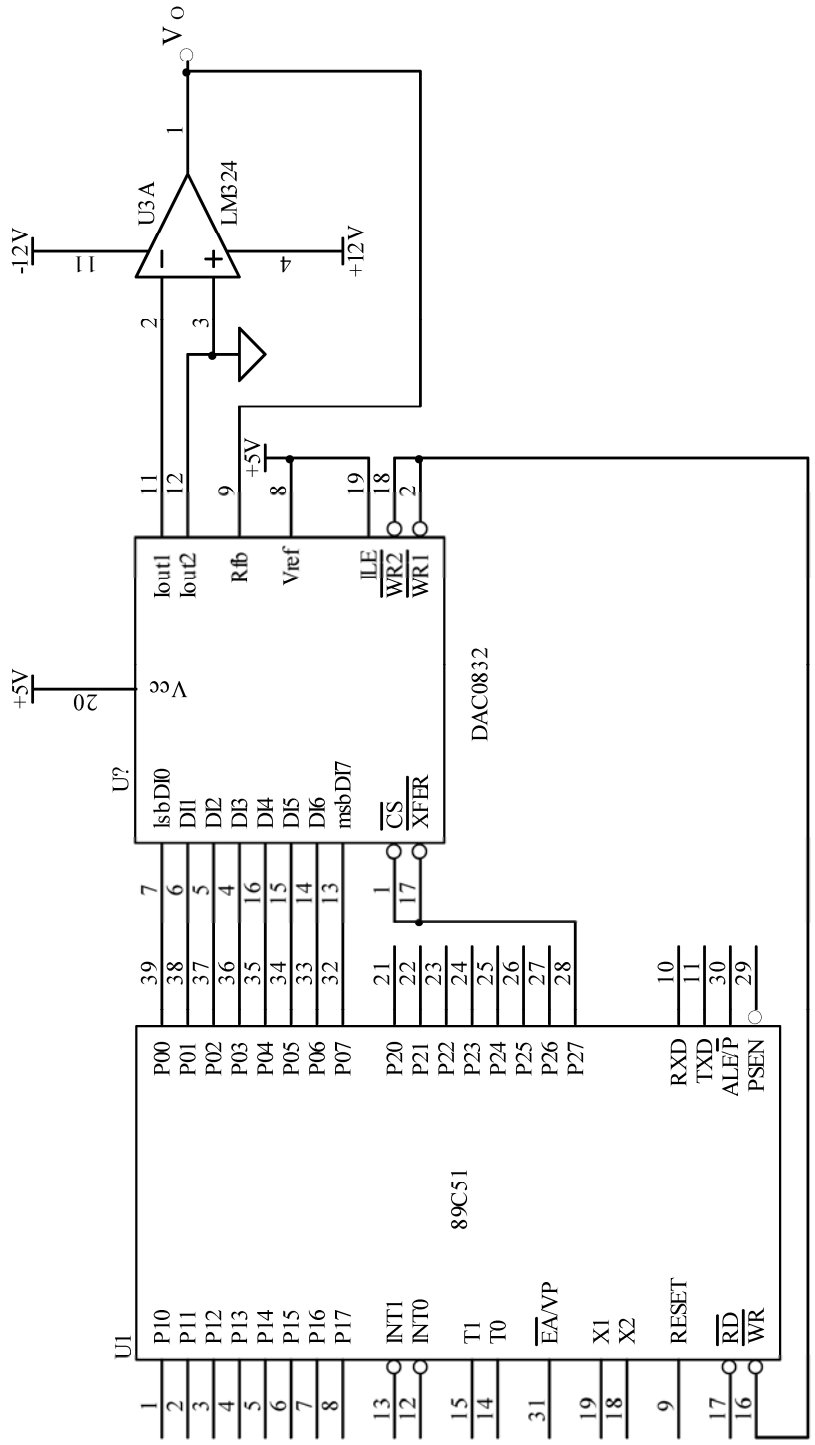
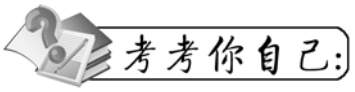


图 10-4　DAC0832 与单片机的接口电路

10.6 如何设计 D/A 转换器与单片机接口程序

单片机执行以下程序，将在运算放大器 U3A 输出端得到一个锯齿波。

```
        ORG 0000H              ；锯齿波信号发生器
START:  MOV DPTR，#7FFFH       ；送 DAC0832 地址
START:  MOV A，#00H            ；输入初始量 00H 到累加器 A
LOOP:   MOVX @DPTR，A          ；输出对应 A 内容的模拟量
        INC A                  ；A 的内容加 1
        SJMP LOOP
        END
```



1. D/A 转换器有何作用？为什么要进行 D/A 转换？
2. 简述 D/A 转换原理。
3. D/A 转换器的主要性能指标有哪些？并简述其含义。
4. 在 D/A 转换器中，要输出矩形波，应如何修改程序？
5. 在 D/A 转换器中，要输出三角波，应如何修改程序？

项目十一　串行通信

——串行端口通信原理及应用

教学目的

掌握：串行端口通信电路设计方法。

理解：串行端口工作方式；串行端口特殊功能寄存器。

了解：异步通信和同步通信。

愿你知多点：

一般情况下，中小规模的单片机应用系统只需要在最小系统的基础上，用串行扩展就能满足应用系统的需要。串行扩展能够最大程度发挥最小系统的资源功能，简化连接线路，只需 1-4 根信号线，结构紧凑，面积小，扩展性好，系统容易修改，可简化系统的设计。在这一章中，我们将通过完成串行通信这一任务，来学习单片机串行端口通信的相关知识。图 11-1 所示为串行端口通信应用实例。

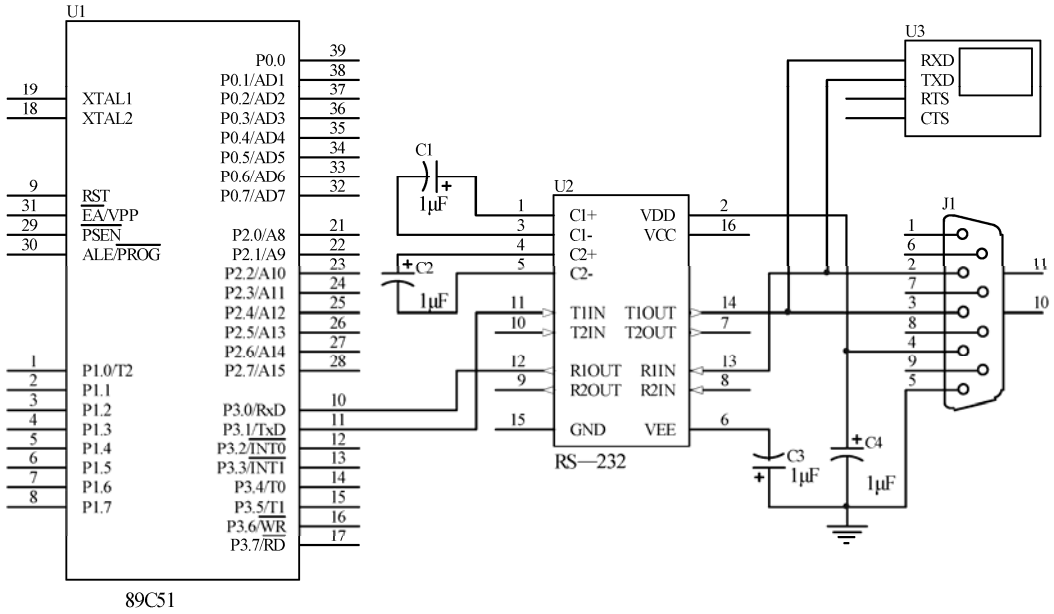


图 11-1　串行端口通信应用实例

11.1 能力培养

本章通过完成串行通信这一任务，可以培养以下能力。

1. 能识别单片机串行端口。
2 能正确使用 74LS164 芯片和 74LS165 芯片。
3 能制作串入并出和并入串出电路，并能编写它们的程序。

11.2 任务分析

要完成此项任务，需要掌握以下 4 方面知识。

1. 如何使用串行端口通信技术。
2. 如何使用 MCS—51 系列单片机串行端口。
3. 如何设计单片机串行端口通信电路。
4. 如何设计单片机串行端口通信程序。

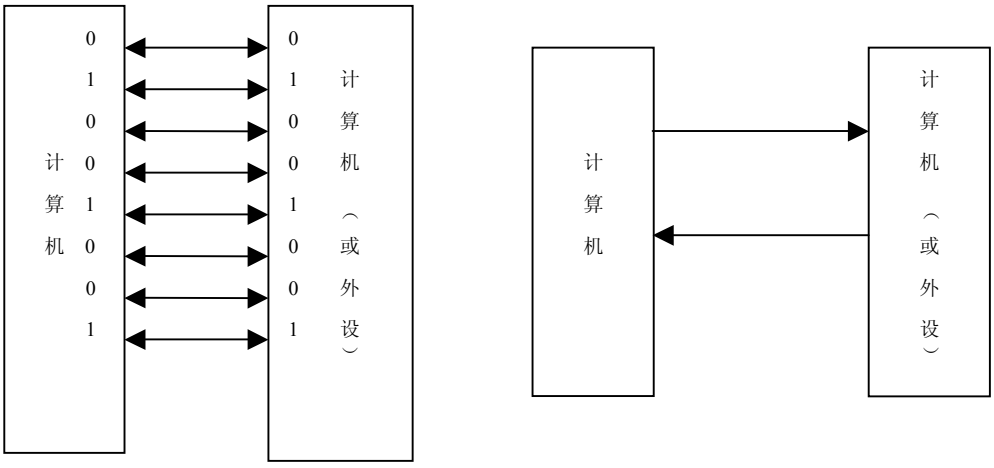
11.3 如何使用串行端口通信技术

所谓“通信”是指计算机与其他设备之间进行的信息交换。通信的方式分为并行通信和串行通信两种。

并行通信是构成一组数据的各位同时进行传送，例如 8 位数据或 16 位数据并行传送。其特点是传输速度快，但当距离较远、位数又多时就会导致通信线路复杂，且并行通信的成本高。

串行通信是数据一位接一位地顺序传送。其优点是通信线路简单，只要一对传输线就可以实现通信（如电话线），可大大地降低成本，适用于远距离通信，缺点是传送速度慢。

图 11-2 所示为并行通信方式及串行通信方式示意图。其中图 11-2（a）所示为并行通信方式；图 11-2（b）所示为串行通信方式。由图可知，假设并行传送 N 位数据所需时间为 T，那么串行传送的时间至少为 NT，但实际上串行传送的时间总是大于 NT 的。



（a）并行通信方式　　（b）串行通信方式

图 11-2　并行通信方式及串行通信方式示意图

11.3.1　串行通信的分类

串行通信按同步方式可分为异步通信和同步通信。

同步通信依靠同步字符保持通信同步。同步通信是由 1～2 个同步字符和多字节数据位组成，同步字符作为起始位以触发同步时钟信号开始发送或接收数据；多字节数据之间不允许有空隙，每位占用的时间相等；空闲位需发送同步字符。同步通信传输速度较快，但要求有准确的时钟信号来实现收发双方的严格同步，对硬件要求较高，适用于成批数据传送。

异步通信依靠起始位与停止位保持通信同步。异步通信数据传送按帧传输，一帧数据包含起始位、数据位、校验位和停止位。异步通信对硬件要求较低，实现起来比较简单、灵活，适用于数据的随机发送与接收，但因每个字节都要建立一次同步，即每个字符都要额外附加两位，所以工作速度较低。在单片机异步通信中，数据分为一帧一帧地传送，即异步串行通信一次传送一个完整字符，其字符格式如图 11-3 所示。一个字符应包括以下几点。

1.　起始位

对应逻辑 0（space）状态。发送器通过发送起始位开始一帧字符的传送。

2.　数据位

在起始位之后传送数据位。数据位中低位在前，高位在后。数据位可以是 5、6、7、8 位。

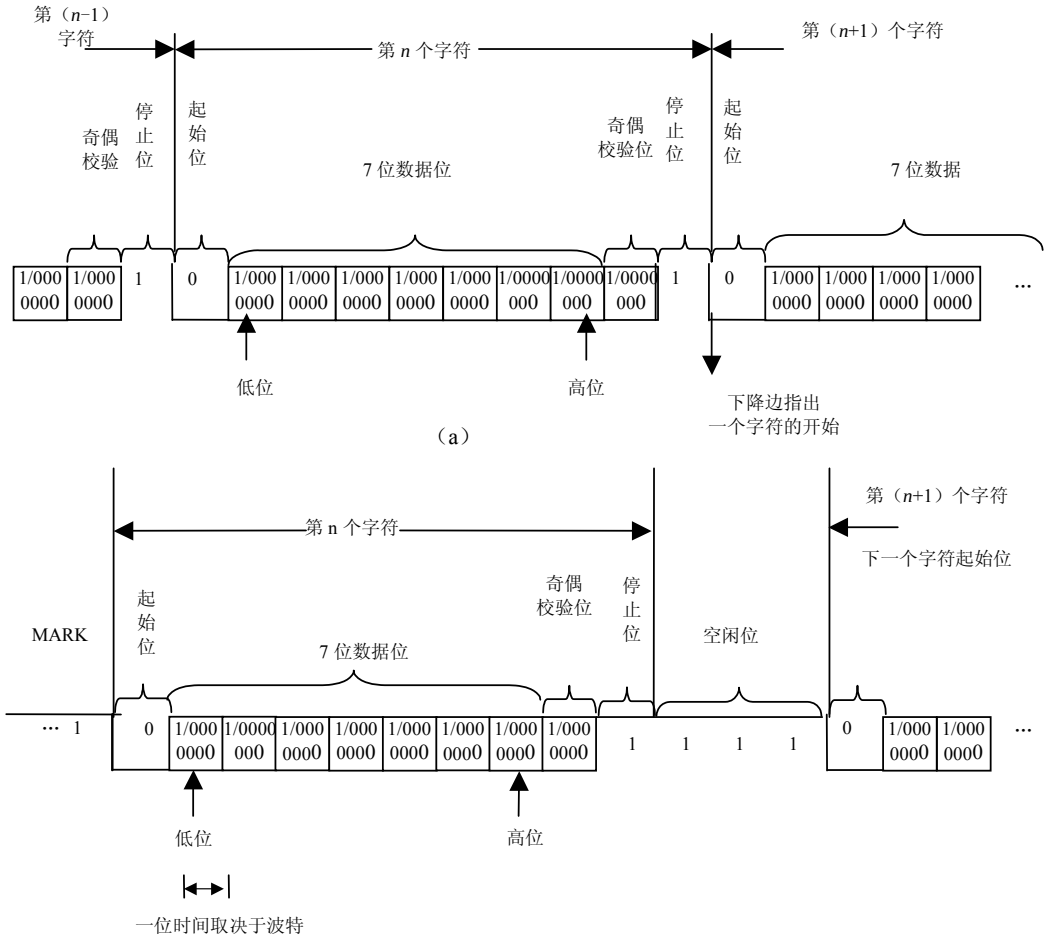


图 11-3　异步串行通信的字符格式

3. 奇偶校验位

奇偶校验位实际上是传送的附加位，若该位用于奇偶校验，可校检串行传送的正确性。奇偶校验位的设置与否及校验方式（奇校验还是偶校验）由用户需要确定。

4. 停止位

停止位可以用逻辑 1（MARK）表示，标志一个字符传送的结束。停止位可以是 1、1.5 或 2 位。

串行通信中用每秒传送二进制数据位的数量表示传送速率，称为波特率。

1 波特＝1bps（位/秒）

例如数据传送速率是 240 帧/秒，每帧由一位起始位、八位数据位和一位停止位组成，则传送速率为

$$10\times240=2400\text{ 位/秒}=2400\text{ 波特}$$

温馨提示：

相互通信的甲乙双方必须具有相同的波特率，否则无法成功地完成串行数据通信。波特率越高，传送速度越快。

11.3.2　串行通信的制式

单片机的串行通信主要采用异步通信传送方式。在串行通信中，按不同的通信方式有单工传送、单双工传送和全双工传送，图 11-4 所示为单片机串行通信方式示意图。

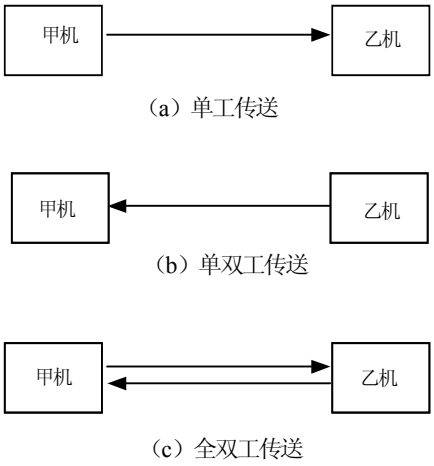


图 11-4　单片机串行通信方式示意图

单工传送中甲乙两机只能单方向发送或接收数据，单双工传送中，甲机和乙机能分时进行双向发送和接收数据，全双工传送时甲乙两机能同时双向发送和接收数据。

11.4　如何使用 MCS—51 系列单片机串行端口

MCS—51 系列单片机有一个全双工的串行端口，这个端口既可以用于网络通信，也可以实现串行异步通信，还可以作为同步移位寄存器使用。

11.4.1　串行端口特殊功能寄存器

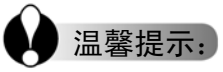
1.　串行数据缓冲器 SBUF

在逻辑上只有一个，既表示发送寄存器，又表示接收寄存器，具有同一个单元地址 99H，用同一寄存器名 SBUF。在物理上有两个，一个是发送缓冲寄存器，另一个是接收缓冲寄存器。发送时，只需将发送数据输入 SBUF，CPU 将自动启动和完成串行数据的发送；接收时，CPU 将自动把接收到的数据存入 SBUF，用户只需从 SBUF 中读出接收数据。程序

如下所示。

```
MOV  SBUF，A    ；CPU 写 SBUF，就是修改发送缓冲器
MOV  A，SBUF    ；CPU 读 SBUF，接收缓冲器数据
```

串行端口对外也有两条独立的收发信号线 RXD(P3.0)和 TXD(P3.1)，因此可以同时发送、接收数据，实现全双工传送。



发送和接收过程都是在发送和接收时钟控制下进行的，必须与设定的波特率保持一致。一般 MCS—51 系列单片机的串行端口时钟是由内部定时器的溢出率经 16 分频后提供的。

2. 串行控制寄存器 SCON

SCON 用来控制串行端口的工作方式和状态（可位寻址）。在复位时所有位被清零，字地址为 98H。SCON 的格式和内容见表 11-1 所列。

表 11-1　SCON 的格式和内容

SCON	D7	D6	D5	D4	D3	D2	D1	D0
位名称	SM0	SM1	SM2	REN	TB8	RB8	TI	RI
位地址	9FH	9EH	9DH	9CH	9BH	9AH	99H	98H
功能	工作方式选择		多机通信控制	接收允许	发送第 9 位	接收第 9 位	发送中断	接收中断

（1） SM0 SM1

SM0 SM1 为串行端口工作方式选择位，其工作方式、功能及波特率见表 11-2 所列。

表 11-2　SM0 SM1 的工作方式、功能及波特率

SM0 SM1	工作方式	功能简述	波 特 率
0　0	方式 0	8 位同步移位寄存器	*f*osc/12
0　1	方式 1	10 位 UART	可变
1　0	方式 2	11 位 UART	*f*osc/12 或 *f*osc/64
1　1	方式 3	11 位 UART	可变

（2） SM2

SM2 为多机通信控制位。当串行端口以方式 2 或方式 3 接收时，如 SM2＝1，则只有当接收到的第 9 位数据（RB8）为 1 时，才将接收到的前 8 位数据送入接收 SBUF，并使 RI 位置 1，产生中断请求信号；否则将接收到的前 8 位数据丢弃。而当 SM2＝0 时，则不论第 9 位数据为 0 还是为 1，都将前 8 位数据装入接收 SBUF 中，并产生中断请求信号。

对方式 0，SM2 必须为 0，对方式 1，当 SM2=1 时，只有接收到有效停止位后才使 RI

位置 1。

（3） REN

REN 为允许接收控制位。当 REN＝0 时，禁止接收；当 REN＝1 时，允许接收。该位由软件置 1 或清零。

（4） TB8

TB8 为方式 2 和方式 3 中要发送的第 9 位数据。

（5） RB8

RB8 为方式 2 和方式 3 中要接收的第 9 位数据。

（6） TI

TI 为发送中断标志。当为方式 0 时，发送完第 8 位数据后，该位由硬件置位。在其他方式下，于发送停止位之前由硬件置位。因此 TI＝1，表示帧发送结束。其状态既可供软件查询使用，也可请求中断。TI 位由软件清零。

（7） RI

RI 为接收中断标志。当为方式 0 时，接收完第 8 位数据后，该位由硬件置 1。在其他方式下，当接收到停止位时，该位由硬件置位。因此 RI＝1，表示帧接收结束。其状态既可供软件查询使用，也可以请求中断。RI 位由软件清零。

3. 电源控制寄存器 PCON

PCON 主要是为 CHMOS 型单片机的电源控制而设置的专用寄存器，单元地址为 87H，不能位寻址。PCON 中 SMOD 位（D7 位即最高位）可影响串行端口的波特率。当（SMOD）＝1，串行端口波特率加倍。（在 PCON 中只有这一个位与串行端口有关）

4. 中断允许寄存器 IE

IE 中 ES 位可选择串行端口中断允许或禁止。

当 ES=0 时，禁止串行端口中断。

当 ES=1 时，允许串行端口中断。

11.4.2 串行行端口的工作方式

80C51 串行通信共有 4 种工作方式，由串行控制寄存器 SCON 中 SM0 SM1 决定，见表 11-3 所列。

1. 工作方式 0

为同步移位寄存器方式，其波特率是固定的，为 *f*osc（振荡频率）的 1/12。

方式 0 发送：数据从 RXD 引脚串行输出，TXD 引脚输出同步脉冲信号，如图 11-5 所示。当 1 个数据写入串行端口发送缓冲器时，串行端口将 8 位数据以 *f*osc/12 的固定波特率从 RXD 引脚输出，从低位到高位。发送完后置中断标志 TI 为 1，呈中断请求状态，在再次发送数据之前，必须用软件将 TI 清零。

表 11-3　串行通信工作方式选择

SM0　SM1	工作方式	说　　明	波特率
0　0	方式 0 （扩展 I/O 口）	移位输入/输出 （用于扩展 I/O 引脚）方式	为 fosc（振荡频率）的 1/12。
0　1	方式 1 （常用）	波特率可变的 8 位 异步串行通信方式	$\frac{T1\text{溢出率}\cdot 2^{SMOD}}{32}$
1　0	方式 2 （不常用）	波特率固定的 9 位 异步串行通信方式	$\frac{f_{OSC}\cdot 2^{SMOD}}{64}$
1　1	方式 3 （常用）	波特率可变的 9 位 异步串行通信方式	$\frac{T1\text{溢出率}2^{SMOD}}{32}$

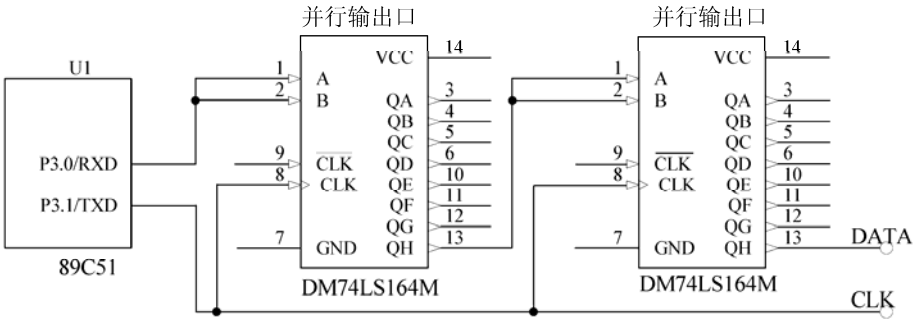


图 11-5　方式 0 发送

方式 0 接收：在满足 REN=1（允许接收）、RI=0 的条件下，串行端口处于方式 0 输入，如图 11-6 所示。此时，RXD 为数据输入端，TXD 为同步信号输出端，接收器也以 *f*osc/12 的波特率采样 RXD 引脚输入的数据信息。当接收器接收完 8 位数据后，置中断标志 RI=1 为请求中断，在再次接收之前，必须用软件将 RI 清零。

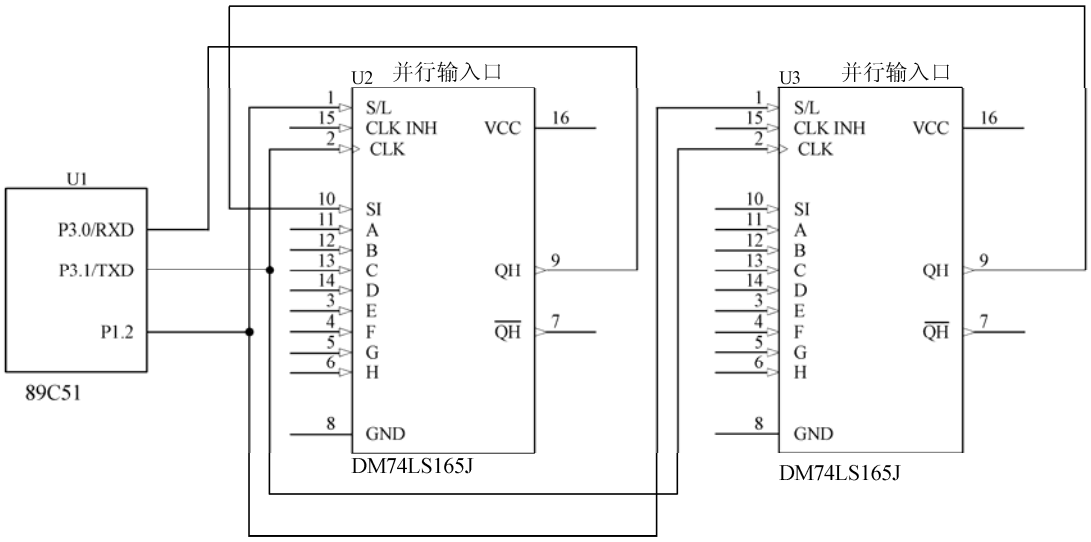


图 11-6　方式 0 接收

温馨提示：

在方式 0 工作时，必须使 SCON 寄存器中的 SM2 位为“0”，但这并不影响 TB8 位和 RB8 位。方式 0 发送或接收完 8 位数据后由硬件置位 TI 或 RI 中断请求标志，CPU 在响应中断后要用软件清除 TI 或 RI 标志。若串行端口要作为并行口输入或输出，这时必须设置串入并出或并入串出的移位寄存器来配合使用（如 74HC164 或 74HC165 等）。

2. 工作方式 1

该方式为波特率可变的 8 位异步通信接口。

方式 1 发送：数据位由 TXD 端输出，发送 1 帧信息为 10 位，其中 1 位起始位、8 位数据位（先低位后高位）和一个停止位“1”。CPU 执行 1 条数据写入发送缓冲器 SBUF 的指令，就启动发送器发送。当发送完数据，就置中断标志 TI 为 1。方式 1 所传送的波特率取决于定时器 T1 的溢出率和特殊功能寄存器 PCON 中 SMOD 的值，即方式 1 的波特率=（2SMOD/32）×定时器 T1 的溢出率。

方式 1 接收：当串行端口置为方式 1，且 REN=1 时，串行端口处于方式 1 输入状态。它以所选波特率的 16 倍的速率采样 RXD 引脚状态。

3. 工作方式 2

该方式为 11 位异步通信接口。

方式 2 发送：发送数据由 TXD 端输出，发送 1 帧信息为 11 位，其中 1 位起始位（0）、8 位数据位（先低位后高位）、1 位可控位为 1 或 0 的第 9 位数据、1 位停止位。附加的第 9 位数据为 SCON 中的 TB8，它由软件置位或清零，可作为多机通信中地址/数据信息的标志位，也可作为数据的奇偶校验位。

方式 2 中使用 TB8 作为发送数据的奇偶校验位，发送程序如下所示。

```
PIPL:   PUSH    PSW       ；保护现场
        PUSH    ACC
        CLR TI            ；清零发送中断标志
        MOV A,@R0         ；取数据
        MOV C,P           ；奇偶位送 C
        MOV TB8,C         ；奇偶位送 TB8
        MOV SBUF,A        ；数据写入发送缓冲器,启动发送
        INC R0            ；数据指针加 1
        POP ACC           ；恢复现场
        POP PSW
        RETI              ；中断返回
```

方式 2 接收：当串行端口置为方式 2，且 REN=1 时，串行端口以方式 2 接收数据。方式 2 的接收方法与方式 1 基本相似。数据由 RXD 端输入，接收 11 位信息，其中 1 位

起始位（0）、8 位数据位、1 位附加的第 9 位数据、1 位停止位（1）。方式 2 的波特率＝（2SMOD/64）×*f*osc。

若附加的第 9 位数据为奇偶校验位，在接收中断服务程序中应作检验处理，参考程序如下所示。

方式 2 中使用 TB8 作为发送数据的奇偶校验位，接收程序如下所示。

```
PIPL:   PUSH    PSW       ; 保护现场
        PUSH    ACC
        CLR RI            ; 清零接收中断标志
        MOV A,SUBF        ; 接收数据
        MOV C,P           ; 取奇偶校验位
        JNC L1            ; 偶校验时转 L1
        JNB RB8,ERR       ; 奇校验时 RB8 为 0 转出错处理
        SJMP    L2
L1:     JB  RB8,ERR       ; 偶校验时 RB8 为 1 转出错处理
L2:     MOV @R0,A         ; 奇偶校验对时存入数据
        INC R0            ; 修改指针
        POP ACC           ; 恢复现场
        POP PSW
        RETI              ; 中断返回
ERR:    …                 ; 出错处理
        RETI              ; 中断返回
```

4. 工作方式 3

方式 3 为波特率可变的 9 位异步通信方式，除了波特率有所区别之外，其余方式都与方式 2 相同。方式 3 的波特率＝（2SMOD/32）×（定时器 T1 的溢出率）。

11.5 如何设计单片机串行端口通信电路

图 11-7 所示为芯片 74LS164 与单片机组成的串入并出接口电路，串行端口作为并行输出端口使用时，要有串入并出的移位寄存器配合（例如 74LS164），在移位时钟脉冲信号（TXD）的控制下，数据从串行端口 RXD 端逐位移入 74LC164 芯片的第 8 脚。当 8 位数据全部移出后，SCON 寄存器的 TI 位被自动置 1，随后 74LS164 的内容即可并行输出。

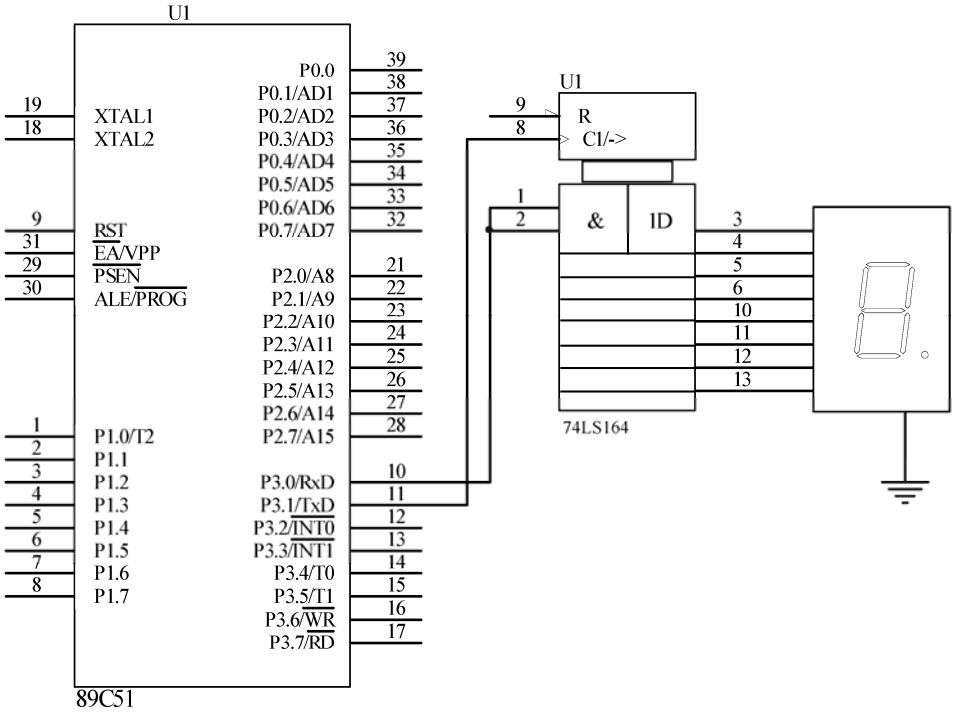


图 11-7　芯片 74LS164 与单片机组成的串入并出接口电路

温馨提示：

这种显示电路的特点是数据串入并出，所谓 “串入并出”，即“串入”是指数据一位一位移入 74LS164 芯片，“并出”是指经过 74LS164 芯片的内容整顿停留后数据一并送出，在本任务中，数据一起送到数码管中，驱动数码管显示字型。

图 11-8 所示为芯片 74LS165 与单片机组成的并入串出接口电路，把能实现“并入串出”功能的移位寄存器（例如 74LS165）与串行端口配合使用，就可以把串行端口变为并行输入端口使用。芯片 74LS165 的 SH/$\overline{\text{LD}}$ 端为移位控制/置入控制（低电平有效）端，当 SH/$\overline{\text{LD}}$ =0 时，从 D0～D7（即 74LS165 芯片的 11 脚～14 脚和 3～6 脚）并行置入数据，当 SH/$\overline{\text{LD}}$ =1 时，允许从 SO 端（即 74LS165 芯片的 9 脚）移出数据。当 SCON 中的 REN=1 时，TXD 端发出移位时钟脉冲信号，从 RXD 端串行输入 8 位数据。当接收到第 8 位数据 D7 后，置位中断标志位 RI，表示一帧数据接收完成。

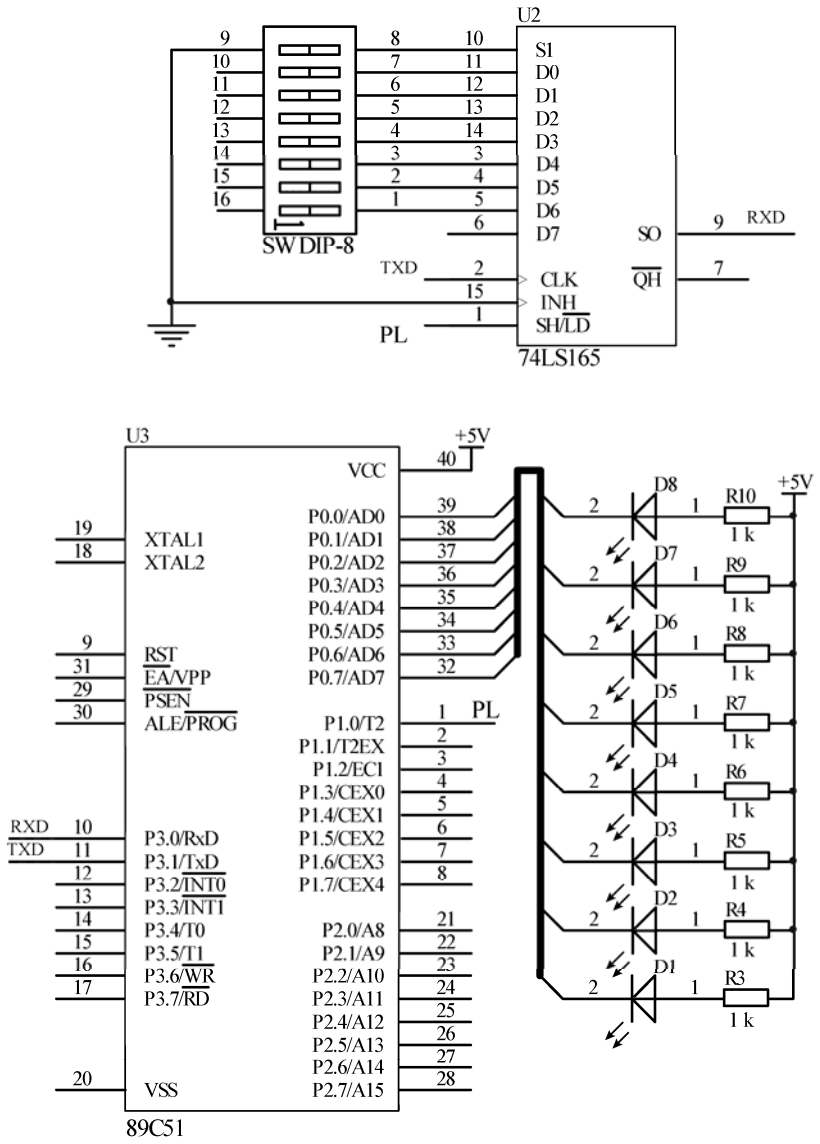


图 11-8　芯片 74LS165 与单片机组成的并入串出接口电路

温馨提示：

这种显示电路的特点是数据串入并出，所谓的“并入串出”，即“并入”是指外部数据从 D0～D7 端并行输入到 74LS165 芯片中，“串出”是指数据经过 74HC164 芯片的内容整顿停留后，通过 SO 端一位一位送到 89C51 芯片的 P 3.0 脚。

11.6　如何设计单片机串行端口通信程序

11.6.1　任务分析

单片机的应用系统由硬件和软件组成，上述硬件原理图搭建完成上电之后，我们还不能看到单片机串行端口通信的现象，还需要编写程序控制单片机管脚电平的高低变化使单片机工作，以实现对 74LS164 和 74LS165 芯片的控制。软件编程是单片机应用系统中的一个重要的组成部分，是单片机学习的重点和难点。

1. 如何实现 74LS164 与单片机组成的串入并出

数据从 RXD 引脚串行输出，TXD 引脚输出同步脉冲信号。当 1 个数据写入串行端口发送缓冲器时，串行端口将 8 位数据以 *f*osc/12 的固定波特率从 RXD 引脚输出，从低位到高位。发送完后置中断标志 TI 为 1，呈中断请求状态，在再次发送数据之前，必须用软件将 TI 清零。

2. 如何实现 74LS165 与单片机组成的并入串出

在满足 REN=1（允许接收）、RI=0 的条件下，串行端口处于方式 0 输入。此时，RXD 为数据输入端，TXD 为同步信号输出端，接收器也以 *f*osc/12 的波特率采样 RXD 引脚输入的数据信息。当接收器接收完 8 位数据后，置中断标志 RI=1 为请求中断，在再次接收之前，必须用软件将 RI 清零。

11.6.2　程序流程图设计

芯片 74LS164 与单片机组成的串入并出接口电路程序流程图如图 11-9 所示。

芯片 74LS165 与单片机组成的并入串出接口电路程序流程图如图 11-10 所示。

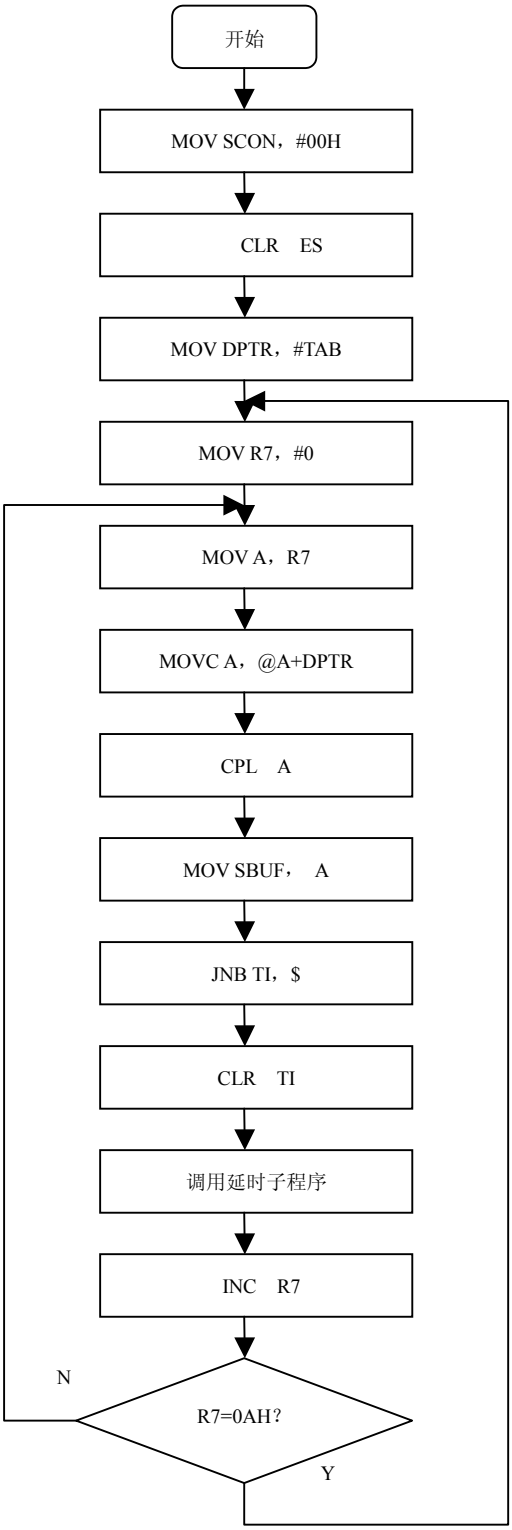


图 11-9　芯片 74LS164 与单片机组成的串入并出接口电路程序流程图

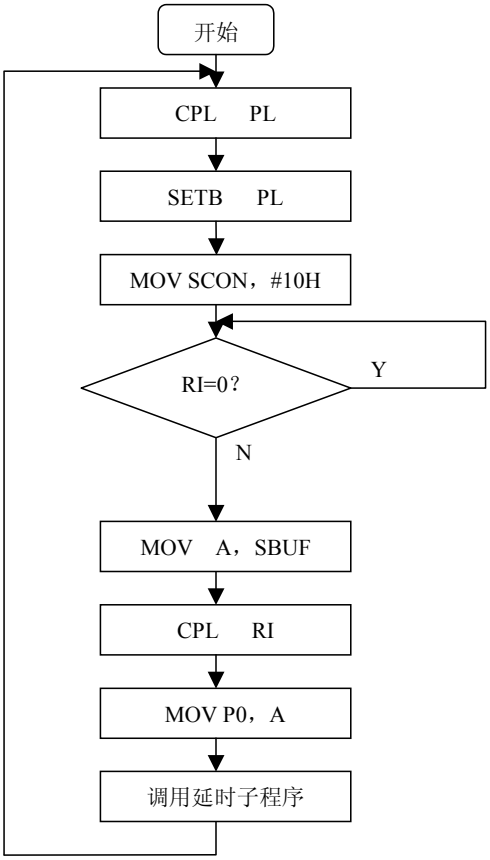


图 11-10　芯片 74LS165 与单片机组成的并入串出接口电路程序流程图

11.6.3　程序清单

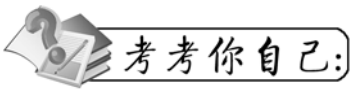
程序一：芯片 74LS164 与单片机组成的串入并出接口电路的程序

```
        ORG  0000H
        AJMP  LIGHT
LIGHT:  MOV SCON，#00H          ；设置串行端口为方式 0
        CLR  ES                 ；ES=0，禁止串行中断
        MOV DPTR，#TAB          ；把 DPTR 指针指向数据表首地址
LOOP:   MOV R7，#00H
LP1:    MOV A，R7
        MOVC A，@A+DPTR         ；基址+变址再把内容送 A
        CPL  A
        MOV  SBUF，A            ；启动串行发送
        NB  TI，$               ；等待发送完毕
        CLR  TI                 ；TI=0，清发送中断标志位
        LCALL  DYS              ；调用延时子程序
        INC  R7                 ；R7 加 1，为指向下一个控制字做准备
```

```
        CJNE  R7，#0AH，LP1      ；判断循环完否？未完继续
        SJMP  LOOP               ；数码管 0～9 已经显示一遍，再从 0 开始重新循环
DYS：   MOV  R4，#10
DD1：   MOV  R6，#0FFH
DD2：   MOV  R5，#0FFH
        DJNZ  R5，$
        DJNZ  R6，DD2
        DJNZ  R4，DD1
        RET
TAB：   DB 03H，9FH，25H，0DH，99H，49H，41H，1FH，01H，09H
        END
```

程序二：芯片 74LS165 与单片机组成的并入串出接口电路的程序

```
        PL   BIT  P1.0          ；把 P1.0 的位地址赋给字符 PL
        ORG  0000H
START： CLR  PL                 ；PL=0，从 D0～D7 端并行输入数据，现在数据可以从外部
                                ；输入到芯片 74LS165 中
        SETB  PL                ；PL=1，发送移位脉冲，允许移出数据
        MOV  SCON，#10H         ；设置串行端口为方式 0，允许串行端口接收数据（REN=1）
WAIT：  JNB  RI，WAIT           ；等待接收完毕。
        MOV  A，SBUF            ；读取数据，把数据暂存在 A 中
        CLR  RI                 ；清除接收中断标志
        MOV  P0，A              ；接收到的数据送 P0 端口显示
        ACALL  DELAY            ；延时子程序
        SJMP  START
DELAY： MOV  R4，#0FFH
AA1：   MOV  R5，#0FFH
AA：    NOP
        NOP
        DJNZ  R5，AA
        DJNZ  R4，AA1
        RET
        END
```



1. 串行数据缓冲器 SBUF 有什么作用？
2. 简述串行控制寄存器 SCON 各位的名称、含义和功能。
3. 芯片 74LS164 与单片机组成的串入并出接口电路具体如何实现？
4. 芯片 74LS165 与单片机组成的并入串出接口电路具体如何实现？
5. 以 89C51 串行方式 1 为例，说明其串行发送和接收的工作过程。

项目十二　密码锁设计

——I^2C 总线技术

教学目的

掌握：I^2C 总线接口电路设计方法。
理解：I^2C 总线编程方法；I^2C 总线工作原理。
了解：I^2C 总线数据传送的模式。

愿你知多点：

在我们的生活中，经常使用各种 I^2C 总线产品，如 IC 卡、智能仪表、电视等。那么这些 I^2C 总线产品是如何制作的呢？在这一章中，我们将通过完成密码锁设计这一任务来学习 I^2C 总线的相关知识。图 12-1 所示为 I^2C 总线的应用实例。

图 12-1　I^2C 总线的应用实例

12.1　能力培养

本章通过完成密码锁设计这一任务，可以培养以下能力。

1. 能使用 I^2C 总线器件 AT24C02。
2. 能设计电子密码锁电路。
3. 能编写单片机 I^2C 总线数据模拟程序。
4. 能编写电子密码锁程序。

12.2　任务分析

要完成此项任务，需要掌握以下 5 方面知识。

1. 如何使用 I^2C 总线。

2. 如何使用器件 AT24C02。
3. 如何设计电子密码锁电路。
4. 如何编写单片机 I^2C 总线数据模拟程序。
5. 如何编写电子密码锁程序。

12.3 如何使用 I^2C 总线

12.3.1 I^2C 总线

I^2C 总线（Inter Integrate Circuit BUS）全称为芯片间总线，它是由 PHILIPS 公司开发的两线式串行总线，用于连接微控制器及其外围设备。I^2C 总线产生于 20 世纪 80 年代，最初是为音频和视频设备开发的，如今主要在服务器管理中使用，其中包括单个组件状态的通信。例如管理员可对各个组件进行查询，以管理系统的配置或掌握组件的功能状态，如电源和系统风扇。可随时监控内存、硬盘、网络、系统温度等多个参数，增加了系统的安全性，方便了管理。I^2C 总线用两根线实现全双工同步数据传送，这是利用 I^2C 总线设计单片机系统时，连线少、可靠性高、成本低的特点，且 I^2C 总线外围器件不需要片选信号，支持热插拔。I^2C 总线单片机的系统结构如图 12-2 所示。

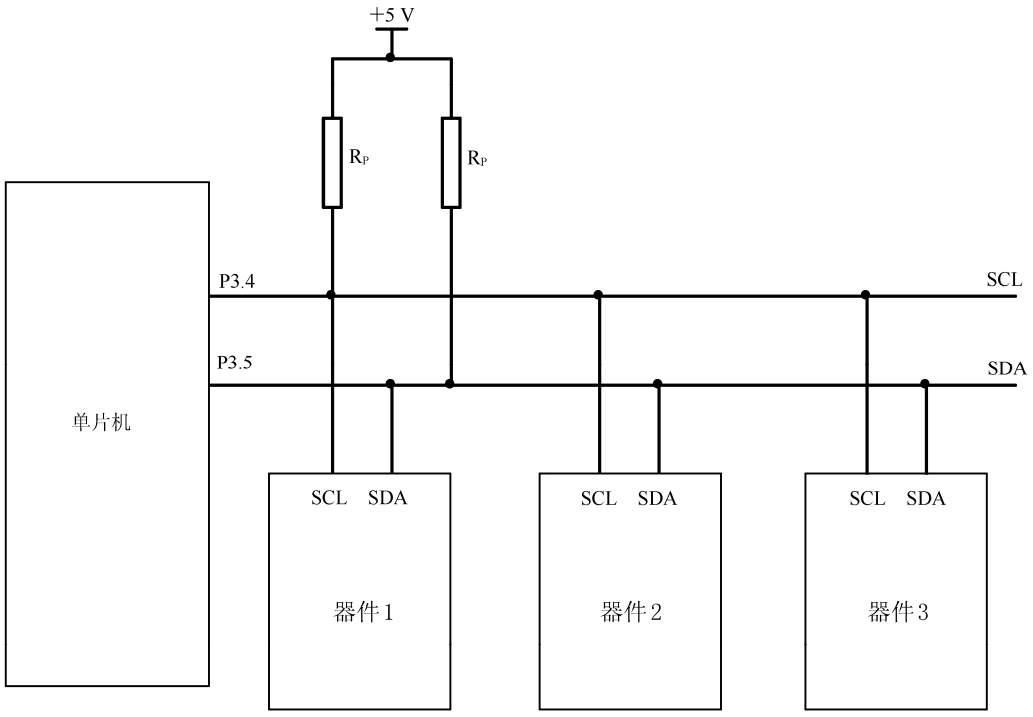


图 12-2 I^2C 总线单片机的系统结构

图中 SDA 是数据线，SCL 是时钟线，由于总线上的各节点都采用漏极开路结构与总线

相连，因此在 SCL 和 SDA 上都需接上拉电阻器。从图中可以看出，I²C 总线系统中的外接元器件都采用线“与”连接方式，这样总线在空闲状态下都保持高电平。I²C 总线在标准模式下数据传送率可达 100 kb/s，高模式下可达 400 kb/s。

I²C 总线有主从和多主两种工作方式。在主从方式中，从器件的地址（包括器件编号地址和引脚地址）由 I²C 总线委员会分配，引脚地址决定引脚外接电平，当器件内部有连续的子地址空间时，对这些空间进行 *N* 个字节的连续读写，子地址会自动加 1。在主从方式的 I²C 总线系统中只需考虑主方式的 I²C 总线操作。本章以主从工作方式为例介绍 I²C 总线的工作过程。

12.3.2　I²C 总线数据传送

I²C 总线上传送的每一个字节均为 8 位，并且高位在前。首先由起始信号启动 I²C 总线，其后为寻址字节，寻址字节由高 7 位地址和最低 1 位方向位组成，方向位表明主控器与被控器数据传送方向，方向位为“0”时表明主控器对被控器进行写操作，为“1”时表明主控器对被控器进行读操作，其后的数据传输字节数是没有限制的，每传送一个字节后都必须跟随一个应答位或非应答位，在全部数据传送结束后主控制器发送终止信号。图 12-3 所示为 I²C 总线的一次完整数据传输过程。

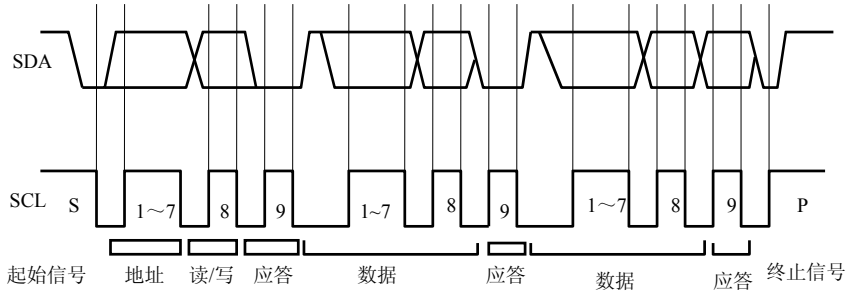


图 12-3　I²C 总线的一次完整数据传输过程

I²C 总线上的数据传输有许多种读/写操作的组合方式。几种常用的数据传送方式如下所示。

读操作：

S	SLAR	A	Data1	A	Data2	A	…	Datan-1	A	Datan	$\overline{A}$	P

写操作：

S	SLAW	A	Data1	A	Data2	A	…	Datan-1	A	Datan	A/$\overline{A}$	P

读/写操作：

S	SLAW/R	A	Data1	A	…	Datan	A/$\overline{A}$	Sr	SLAW/R	A

Data1	A	…	Datan	A/$\overline{A}$	P

温馨提示：

：主控器发送，被控器接收。

：主控器接收，被控器发送。

A：应答信号

$\overline{A}$：非应答信号

S：起始信号

Sr：重复起始信号

P：停止信号

SLAR：读寻址字节

SLAW：写寻址字节

Data1～Datan：被写入/读出的 *n* 个数据字节。

在读/写操作中未注明数据传送方向的，其方向由寻址字节的方向位决定。

12.4 如何使用器件 AT24C02

AT24C02 系列串行 E^2PROM 具有 I^2C 总线接口功能，且功耗小，为宽电源电压（根据不同型号其电压范围在 2.5～6.0 V），工作电流约为 3 mA，静态电流随电源电压不同为 30～110 μA。

目前我国所用的 AT24C02 系列串行 E^2PROM 主要由 ATMEL、NATIONAL、MICROCHIP、XICOR 等几家公司提供。下面以 ATMEL 公司产品为例进行说明。

1. AT24C02 系列串行 E^2PROM 的引脚、容量、结构

AT24C02 系列串行 E^2PROM 目前在我国应用最多的是 8 脚封装，其引脚图如图 12-4 所示。

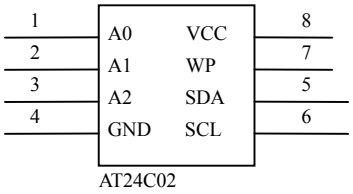


图 12-4 AT24C02 系列串行 E^2PROM 的引脚图

AT24C02 引脚说明如下所示。

A0、A1、A2：片选或页面选择地址输入端，用于对 E^2PROM 的地址编码，A0、A1、A2 组合不同的编码值，可以选中不同的芯片。

SCL：串行时钟信号端，用于控制输入与输出数据的同步。写入串行 E^2PROM 的数据用 SCL 上升沿同步，输出数据用下降沿同步。

SDA：串行数据信号输入/输出端。由于漏极开路结构，因此在使用时该引脚必须接一个约 10 kΩ的上拉电阻器。

WP：写保护，用于硬件数据保护功能。当该引脚接地时，可以对整个存储器进行正常地读/写操作。

VCC：电源电压，+5 V。

GND：地。

2. AT24C02 控制字

所有的串行 E^2PROM 芯片在接收到开始信号后都需要接收一个 8 位含有芯片地址的控制字（又称寻址字节），以确定本芯片是否被选通和将要进行的是读操作还是写操作。控制字格式是高 4 位为器件类型识别符（不同的芯片类型有不同的定义，E^2PROM 一般应为 1010），接着 3 位为片选，也就是 3 个地址位，最后一位为读写控制位，当为 1（Input）时为读操作，当为 0（Output）时为写操作。

12.5　如何设计电子密码锁电路

图 12-5 所示为电子密码锁电路，图中芯片 89C51 为密码识别控制中心，P20、P21、P22 为用户密码输入端，分别接三位密码开关 K1、K2、K3，P23 为固定输出低电平端（由程序实现）；芯片 AT24C02 为 E^2PROM 器件，用于存放预设密码；D1～D8 用于显示密码状态，具体说明如下所示。

D4 亮：提示用户输入密码。

D1 亮：用户已输入第一位密码。

D1、D2 亮：用户已输入第一位和二位密码。

D1～D8 全亮：用户输入的三位密码全正确，开锁。

D1～D8 全灭：密码错误，不开锁。

AT24C02 的器件地址是 1010，A0、A1、A2 为芯片地址位，按图 12-5 的连接方式，芯片地址为 000，因此，AT24C02 在系统中的寻址字节 SLAW=A0H，SLAR=A1H。

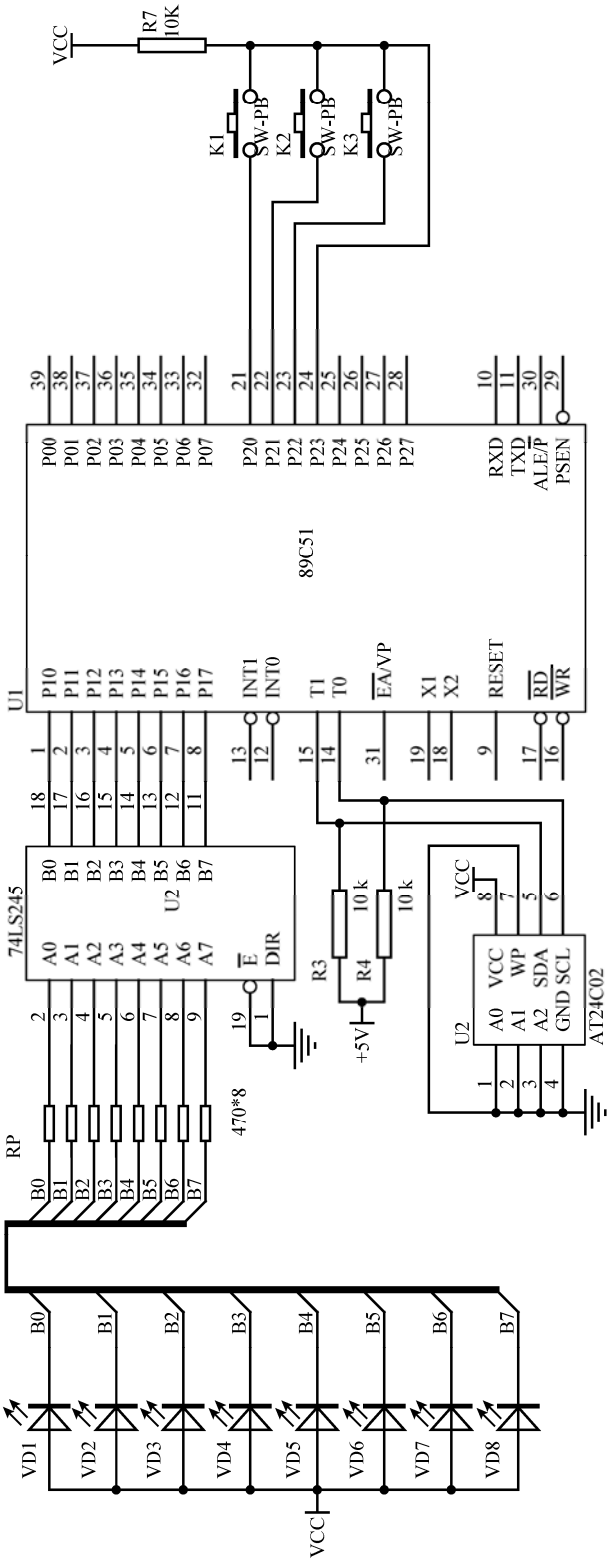


图 12-5　电子密码锁电路

12.6　如何编写单片机 I²C 总线数据模拟程序

在标准 I²C 总线中，总线状态监测由硬件完成，用户无须介入，但是具有 I²C 总线的 MCS—51 系列单片机毕竟是少数，不带 I²C 总线的单片机也其实不必扩展 I²C 总线接口，只要通过软件模拟，就能像具有 I²C 总线的单片机一样使用。前面已经提到，大多数单片机应用系统中只有一个 CPU，这种单主系统如果采用 I²C 总线技术，则总线上只有单片机对 I²C 总线从器件的访问，没有总线的竞争等问题，因此只需要模拟主接收时序和主发送即可。

例如利用 P3.4、P3.5 分别作为时钟线 SCL 和数据线 SDA，电路中晶振为 6 MHz。软件包中包括启动（START）、停止（STOP）、发送应答位（ACK）、发送非应答位（NACK）、应答位检查（CACK）、发送一个字节数据（WRBYT）、接收一个字节数据（RDBYT）、发送 *n* 个字节数据（WRNBYT）、接收 *n* 个字节数据（RDNBYT）9 个子程序，具体程序如下所示。

```
SCL     BIT P3.4
SDA     BIT P3.5
MTD     EQU 40H          ; SMTD 为发送数据缓冲区首址
MRD     EQU 50H          ; SMRD 为接收数据缓冲区首址
SLA     EQU 60H          ; SLA 为寻址字节 SLAR/W 的存放单元
NUMBYT  EQU 6lH          ; NUMBYT 为传送字节数存放单元
                         ; 启动 I²C 总线数据传送子程序
START:  CLR SCL          ; 先将 SCL 置低才改变 SDA，以免误操作
        NOP
        NOP
        SETB SDA         ; 启动 I²C 总线
        SETB SCL
        NOP
        NOP
        CLR SDA
        NOP              ; 起始信号保持 4μ
        NOP
        CLR SCL
        RET
                         ; 停止 I²C 总线数据传送子程序
STOP:   CLR SCL          ; 先将 SCL 置低才改变 SDA，以免误操作
        NOP
        NOP
```

```
        CLR SDA
        SETB SCL
        NOP
        NOP
        SETB SDA
        NOP
        NOP
        CLR SCL
        CLR SDA
        RET
        ；发送应答位子程序
ACK:    CLR SCL             ；先将 SCL 置低才改变 SDA，以免误操作
        NOP
        NOP
        CLR SDA
        SETB SCL
        NOP
        NOP
        CLR SCL
        CLR SDA
        RET
        ；发送非应答位子程序
NACK:   CLR SCL             ；先将 SCL 置低才改变 SDA，以免误操作
        NOP
        NOP
        SETB SDA
        SETB SCL
        NOP
        NOP
        CLR SCL
        CLR SDA
        RET
        ；应答位检查子程序
CACK:   CLR SCL             ；先将 SCL 置低才改变 SDA，以免误操作
        NOP
        NOP
        SETB SDA            ；应答位检查，F0=0，应答正常
        SETB SCL
```

```
        CLR F0
        MOV C，SDA
        JNC CEND
        SETB F0
CEND:   CLR SCL
        RET
        ；发送 1 字节数据子程序
WRBYT:  MOV R1，#08H
WLP:    CLR SCL              ；先将 SCL 置低才改变 SDA，以免误操作
        NOP
        NOP
        RLC A
        MOV SDA，C
        SETB SCL
        NOP
        NOP
        CLR SCL
        DJNZ R1，WLP
        RET
        ；接收 1 字节数据子程序
RDBYT:  MOV R1，#08H
RLP:    CLR SCL              ；先将 SCL 置低才改变 SDA，以免误操作
        NOP
        NOP
        SETB SDA
        SETB SCL
        MOV C，SDA
        MOV A，R2
        RLC A
        MOV R2，A
        CLR SCL
        DJNZ R1，RLP
        RET
                             ；发送 n 字节数据子程序
WRNBYT: MOV R2，NUMBYT       ；传送字节数
        LCALL START
        MOV A，WSLA          ；外围器件寻址字节存放单元
        LCALL WRBYT
```

```
        LCALL CACK
        JB F0，WRNBYT
        MOV R1，#MTD            ；发送数据缓冲区首地址
WRDA：  MOV A，@R1
        LCALL WRBYT
        LCALL CACK
        JB F0，WRNBYT
        INC R1
        DJNZ R2，WRDA
        LCALL STOP
        RET
                                ；接收 n 字节数据子程序
RDNBYT：MOV R2，NUMBYT          ；NUMBYT 为传送字节数
        LCALL，START
        MOV A，VSLA
        LCALL WRBYT
        LCALL CACK
        JB F0，RDNBYT
        MOV R1，#MRD            ；MRD 为接收缓冲区首地址
RDN1：  LCALL RDBYT
        MOV @R1，A
        DJNZ R2，ACK
        LCALL MNAK
        LCALL STOP
        RET
ACK：   LCALL ACK
        INC R1
        SJMP RDN1
```

12.7 如何编写密码锁程序

要编写密码锁程序，必须解决输入密码、识辨密码和提示用户三方面的问题。

12.7.1 程序流程图设计

密码锁程序流程图如图 12-6 所示。程序首先进行初始化，主要用于定义 I^2C 总线的数据与时钟总线、读/写控制字节存放单元、发送/接收数据缓冲区首址以及设置密码与用户密码存放单元等。

然后调用预置密码子程序和输入用户密码子程序，并将两个密码进行比较，若密码一致，则 P1 端口输出低电平，D1～D8 亮，开锁，否则 P1 端口输出高电平，D1～D8 灭。最后调用读取设置密码子程序与识辨密码子程序。

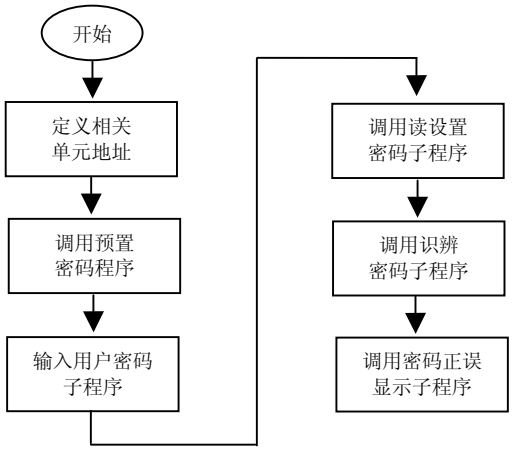


图 12-6　电子密码锁程序流程图

12.7.2　程序清单

```
            ORG 0100H
SCL         BIT P3.4              ；定义时钟总线
SDA         BIT P3.5              ；定义数据总线
SLAW        EQU 30H               ；写控制字节存放单元
SLAR        EQU 31H               ；读控制字节存放单元
MTD         EQU 40H               ；MTD 为发送数据缓冲区首址
MRD         EQU 50H               ；MRD 为接收数据缓冲区首址
SLA         EQU 60H               ；SLA 为寻址字节 SLAR/W 的存放单元
PASSW       EQU 65H               ；预置密码暂放单元首址
NEWPASSW    EQU 67H               ；用户密码存放单元首址
            ORG 0200H
MAIN:       MOV P1，#0FFH         ；关显示
            CALL PREPASSW         ；调用预置密码子程序
            CALL WTPASSW          ；调用预置密码写入子程序
AGAIN:      CALL INPASSW          ；调用用户密码输入子程序
            CALL RDPASSW          ；调用读取预置密码子程序
            CALL CMPPASSW         ；调用识辨密码子程序
            CALL DISP             ；调用密码正误显示子程序
            AJMP AGAIN
```

```
                ；预置密码子程序
PREPASSW:       MOV R0，PASSW           ；取预置密码暂放单元首址
                MOV @R0,#12H            ；预置密码“123”
                INC R0
                MOV @R0,#03H
                MOV R0，PASSW           ；密码保存到发送数据缓冲区
                MOV A,@R0
                MOV R1,MTD
                MOV @R1，A
                INC R0
                INC R1
                MOV A,@R0
                MOV @R1，A
                RET
                ；预置密码写入子程序
WTPASSW:        MOV SLAW，#0A0H         ；控制字
WRADD:          MOV R2，#02H            ；发送 2 字节数据
                LCALL START             ；启动发送
                MOV A，SLAW             ；传送控制字
                LCALL WRBYT             ；调写字节数据子程序
                LCALL CACK              ；调应答位检查子程序
                JB F0，WRADD
                MOV A，#00H             ；写入 E2PROM 单元地址
                LCALL WRBYT
                LCALL CACK
                JB F0，WRADD
                MOV R0，MTD
WRDA:           MOV A，@R0              ；取发送数据缓冲区地址
                LCALL WRBYT
                LCALL CACK
                JB F0，WRADD
                INC R0                  ；修改发送数据缓冲区地址指针
                DJNZ R2,WRDA            ；没有发送完，继续发送
                LCALL STOP              ；调停止子程序
                LCALL DELAY_10 ms       ；调 10 ms 延时子程序
                RET
                ；用户密码输入子程序
INPASSW:        MOV P2,#00H             ；去干扰
```

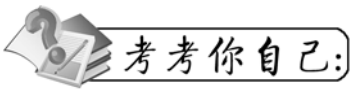
```
            MOV P1，#0F7H          ；点亮用户密码输入指示灯
            MOV R0，NEWPASSW       ；去用户密码首址
            MOV R1，#00H           ；R1 用于存放密码位数
            MOV 20H,#00H
LOOP9:      JNB P2.0，LOOP0        ；K1 按下，转 LOOP0
            JNB P2.1，LOOP1        ；K2 按下，转 LOOP1
            JNB P2.2，LOOP2        ；K3 按下，转 LOOP2
            SJMP LOOP9
EXIT0：     INC R0
            MOV @R0,#01H
            SJMP EXIT
EXIT1：     INC R0
            MOV @R0,# 02H
            SJMP EXIT
EXIT2：     INC R0
            MOV @R0,#03H
            SJMP EXIT
EXIT:       RET
LOOP0：     CALL DELAY_10MS        ；延时去抖动
            JB P2.0，LOOP9
            JNB P2.0,$
            INC R1
            MOV A，R1              ；密码位数暂存在 A 中
            SUBB A，#3
            JZ EXIT0
            MOV A,R1
            JZ PASSW01
            ORL 20H,#01H
            MOV @R0,20H
            SJMP LOOP9
PASSW01:    ORL 20H,#10H
            MOV @R0,20H
            SJMP LOOP9
LOOP1：     CALL DELAY_10MS        ；延时去抖动
            JB P2.1，LOOP9
            JNB P2.1,$
            INC R1
            MOV A，R1              ；密码位数暂存在 A 中
```

```
            SUBB A，#3
            JZ EXIT1
            MOV A,R1
            JZ PASSW11
            ORL 20H,#02H
            MOV @R0,20H
            SJMP LOOP9
PASSW11:    ORL 20H,#20H
            MOV @R0,20H
            SJMP LOOP9
LOOP2：     CALL DELAY_10MS          ；延时去抖动
            JB P2.2，LOOP9
            JNB P2.2,$
            INC R1
            MOV A，R1                ；密码位数暂存在 A 中
            SUBB A，#3
            JZ EXIT2
            MOV A,R1
            JZ PASSW21
            ORL 20H,#03H
            MOV @R0,20H
            SJMP LOOP9
PASSW21:    ORL 20H,#30H
            MOV @R0,20H
            SJMP LOOP9
            ；读取设置密码子程序
RDPASSW：   MOV SLAW，#0A0H          ；写控制字
            MOV SLAR，#0A1H          ；读控制字节
RDADD：     MOV R2，#02H             ；传送两字节数据
            LCALL START              ；启动发送
            MOV A，SLAW              ；发送写控制字节
            LCALL WRBYT              ；调用写入字节数据子程序
            LCALL CACK
            JB F0，RDADD
            MOV A，#00H              ；E2PROM 首地址
            LCALL WRBYT
            LCALL CACK
            JB F0，RDADD
```

```
                LCALL START             ；重新启动读取
                MOV A，SLAR             ；读控制字节
                LCALL WRBYT
                LCALL CACK
                JB F0，RDADD
                MOV R0，MRD             ；存入数据缓冲区地址
RDN：           LCALL RDBY              ；读字节数据
                MOV @R0，A              ；存入读取数据
                DJNZ R2，ACK
                LCALL NACK              ；发送非应答位
                LCALL STOP              ；调停止子程序
                RET
ACK：           LCALL ACK
                INC R0
                SJMP RDN
                                        ；识辨密码子程序
CMPPASSW：      MOV R0，NEWPASSW        ；取用户密码保存首址
                MOV R1,MRD              ；接收数据缓冲区（预置密码存放）
                MOV A，@R0
                SUBB A，@R1             ；比较高位密码
                JNZ NEQU                ；不相同，输出高电平
                INC R0                  ；指向低位密码地址
                INC R1
                MOV A，@R0
                SUBB A，@R1             ；比较低位密码
                JNZ NEQU                ；不相同，输出高电平
                MOV A，#00H             ；相同，输出低电平
                AJMP YEQU
NEQU：          MOV A，#0FFH
YEQU：          RET
                ；密码正误显示子程序
DISP：          MOV P1，A
                CALL DELAY1s            ；调 1 s 延时子程序
                RET
                                        ；10 ms 延时子程序
DELAY_10ms：    MOV R1，#20H            ；若晶振为 6 MHz
D1：            MOV R2,#248
D2：            DJNZ R2,D2
```

```
            DJNZ R1,D1
            RET
            ；1 s 延时子程序
DELAY1s：   MOV R0,#100              ；若晶振频率为 6 MHz
D3:         MOV R1，#20H
D4:         MOV R2,#248
D5:         DJNZ R2,D3
            DJNZ R1,D2
            DJNZ R0,D1
            RET
            END
```

注：I^2C 总线信号模拟子程序与前一小节所列举的程序相同，因篇幅原因这里不再给出。



1. 简述 I^2C 串行总线数据传送过程。
2. 在 I^2C 总线中，为什么要接上拉电阻器？
3. 简述器件 AT24C02 各引脚功能。

项目十三　温度计的设计
——单片机应用系统设计

教学目的

掌握：单片机应用系统设计过程；温度计电路设计方法。
理解：温度计程序。
了解：抗干扰系统。

愿你知多点：

单片机在自动检测、智能仪器仪表、数据采集和家电领域中得到了广泛的应用，但单片机只是一个独立的芯片，只有与某些器件、设备有机地组合在一起，并配有相应的程序，才能构成一个真正的单片机应用系统，完成特定的任务。在这一章中，我们将通过完成温度计的设计这一任务来学习单片机应用系统设计的相关知识。

13.1　能力培养

本章通过完成温度计的设计这一任务，可以培养以下能力。

1. 能正确使用温度传感器。
2. 能制作温度计。
3. 能正确设计抗干扰电路。

13.2　任务分析

要完成此项任务，需要掌握以下 4 方面知识。

1. 如何设计单片机应用系统。
2. 如何设计抗干扰系统。
3. 如何设计温度计电路。
4. 如何设计温度计程序。

13.3 如何设计单片机应用系统

13.3.1 单片机应用系统设计步骤

单片机应用系统设计步骤如图 13-l 所示。首先根据用户提出的具体功能对应用系统进行全面分析，确定各项技术指标，编写设计任务书，再根据任务书设计硬件电路和编写软件，最后进行软/硬件的综合调试。

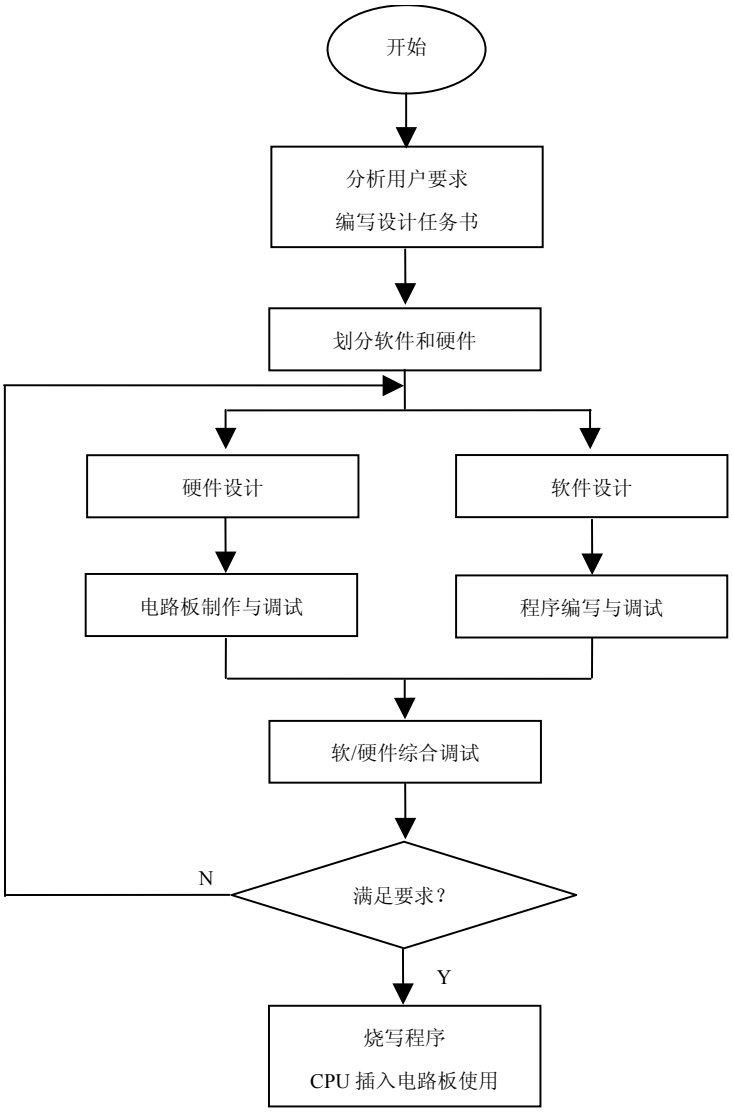


图 13-l 单片机应用系统设计步骤

13.3.2　单片机应用系统硬件开发

硬件设计的任务就是根据设计任务书的要求，首先确定应用系统的总体设计方案，然后再按照总体设计方案要求，选定单片机机型，确定系统中所需要的存储器、I/O 接口电路以及其他外围电路，设计出系统所需要的电路原理图，并根据硬件电路设计印制电路板的电路图。最后将系统中所要使用的元器件焊接在印制电路板上，至此应用系统硬件部分初步完成。

硬件制作完成后，需进行软/硬件综合调试，如果发现问题，则需修改软件或硬件，直至满足用户要求为止。

为了降低开发费用，方便修改电路，在硬件电路较简单的前提下，硬件系统调试可利用面包板完成。

13.3.3　单片机应用系统软件开发

硬件电路是系统的骨架，软件是系统的具体内容，两者结合才能构成一个完整的应用系统。在实际编程时，往往与单片机的输入/输出接口设计、存储器的扩展交织在一起，因此软件开发是系统研制过程中最重要也是最困难的任务，它直接关系到实现系统的功能和性能。

一般来说软件开发可以分为分析任务、确定算法、画程序流程图、编写源程序以及程序调试 5 个步骤，但考虑到实际应用系统的复杂性和多变性，这就要求开发者在实际编程时必须认真考虑每一个步骤，尤其是画出结构清晰、简洁、合理程序流程图。只有这样，在编写各个功能程序时才容易实现模块化、子程序化。

13.4　如何设计抗干扰系统

单片机应用系统的工作环境复杂多变，很容易受到各种干扰的侵袭，特别是在工业环境下工作的单片机系统，恶劣的工业环境往往会给单片机带来各种各样的干扰。因此在进行硬件系统设计和程序设计时，应时刻注意干扰问题。干扰如果解决不好，其他环节设计得再好，整个系统也不可能正常工作。单片机应用系统在工作时常见干扰有电源干扰、过程通道干扰以及电磁干扰等。

13.4.1　电源系统抗干扰措施

电源干扰是单片机应用系统中干扰的主要来源之一，电源干扰包括电源线中的高频干扰、感性负载产生的瞬变噪声、电网电压的短时下降干扰等，对于电源干扰，我们可以采取以下措施。

1.　采用独立电源供电

采用独立电源供电，可将单片机与一些大功率设备分开，对它们分别进行供电，在一

定程度上可以减少干扰。

2. 增加隔离变压器

由于隔离变压器在初级与次级之间增加了一层屏蔽层，并将屏蔽层与铁芯一起接地，这样可以防止干扰通过变压器的初级与次级之间的分布电容进入单片机。

3. 安装退耦电容器

在整流元件上并接小容量退耦电容器，可在很大程度上削弱高频干扰，退耦电容量一般选在 1000 pF～0.01 μF。

13.4.2 过程通道干扰及其抑制

过程通道干扰一般采用隔离和滤波技术来解决，并将数字地和模拟地分开。常用的隔离器件有隔离变压器、光耦器件、继电器和隔离放大器等，其中光耦器件应用最广。

在过程通道中采用光耦器件可将单片机系统与各种传感器、开关、执行机构隔离。光耦器件前后两部分电路应分别采用两组独立电源供电。当数字通道输出的开关量用于控制大负荷设备时，一般不宜用光耦器件，而应采用继电器隔离。

13.5 如何设计温度计电路

温度计电路如图 13-2 所示。其中运算放大器 U6A（LM324J）、电阻器 R6～R9、热敏电阻器 Rt、电位器 W1 和 W2 组成温度采集电路，将温度信号转换成电信号（模拟信号）；8 位 A/D 转换器 U2、地址锁存器 U4、或非门 U5 以及单片机 AT89C51 等组成 A/D 转换器，将模拟信号转换成数字信号；驱动器 U3、数码管 DS0～DS2、三极管 VT1～VT3、排阻 RP 等组成温度计显示电路，将检测后的温度显示出来；电阻器 R4～R5、电容器 C3 和开关 S1 组成单片机的复位电路，为单片机提供复位信号；晶体振荡器 Y、电容器 C1 和 C2 组成单片机的时钟电路，为单片机提供时钟信号；插头 JP1、变压器 T1、整流桥 D1、电容器 C4～C7 组成电源电路，为温度计提供直流电源。

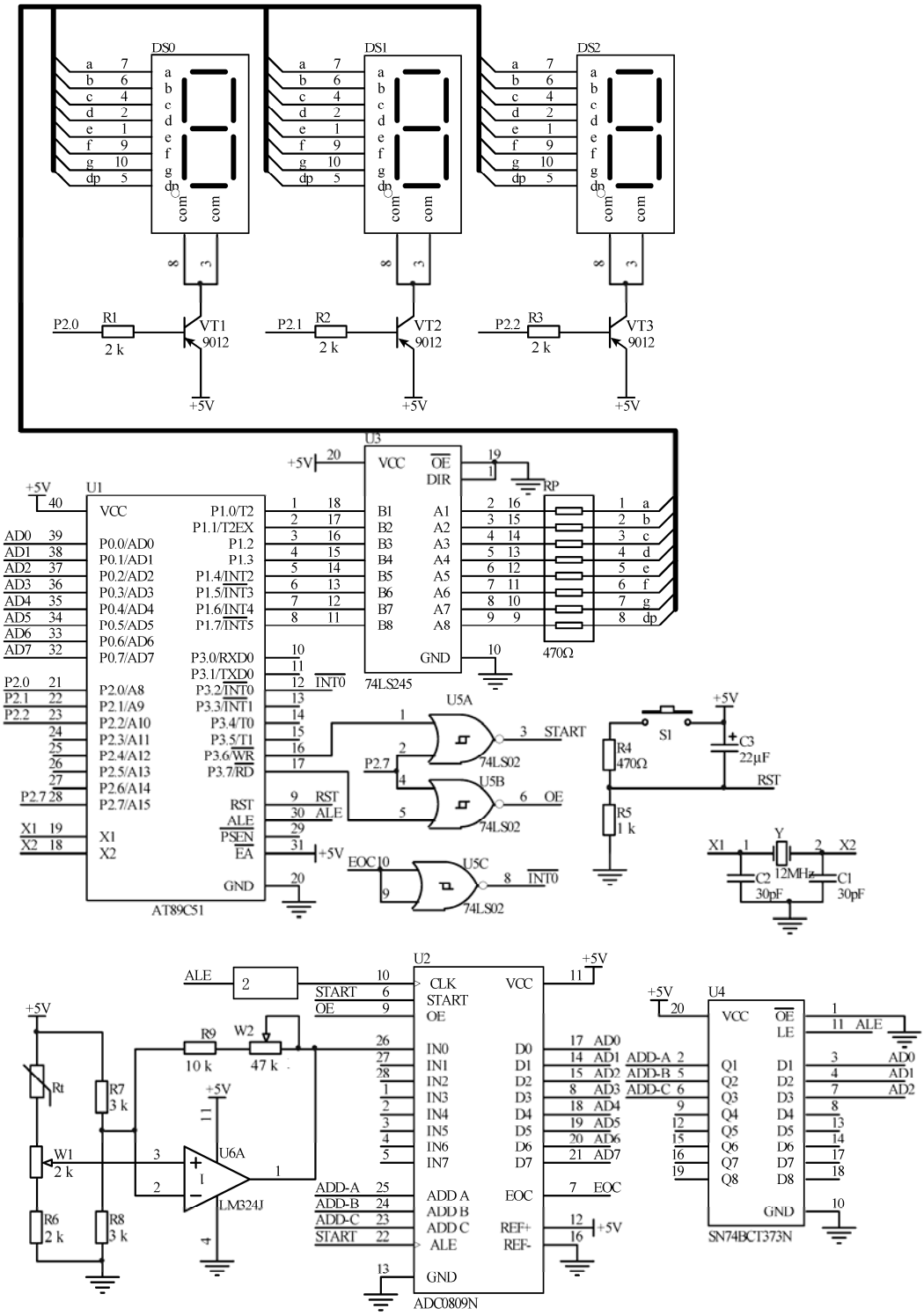

DS0
DS1
DS2
P2.0
P2.1
P2.2
R1
R2
R3
2 k
VT1
VT2
VT3
9012
+5V
U1
AT89C51
U2
ADC0809N
U3
74LS245
U4
SN74BCT373N
U5A
U5B
U5C
74LS02
START
OE
EOC
INT0
RST
ALE
470Ω
S1
C3
22μF
R4
R5
1 k
X1
X2
Y
12MHz
C1
C2
30pF
Rt
R6
R7
R8
R9
W1
W2
10 k
47 k
3 k
U6A
LM324J
ADD-A
ADD-B
ADD-C
AD0
AD1
AD2
AD3
AD4
AD5
AD6
AD7

图 13-2　温度计电路

13.6 如何设计温度计程序

13.6.1 温度计程序流程图

主程序流程图如图 13-3 所示。$\overline{\text{INT0}}$ 中断服务子程序流程图如图 13-4 所示。显示程序流程图如图 13-5 所示。

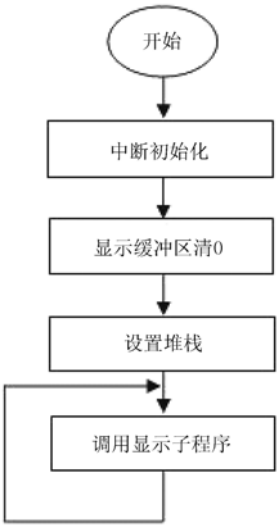


图 13-3 主程序流程图

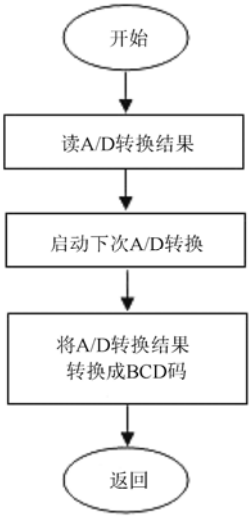


图 13-4 $\overline{\text{INT0}}$ 中断服务子程序流程图

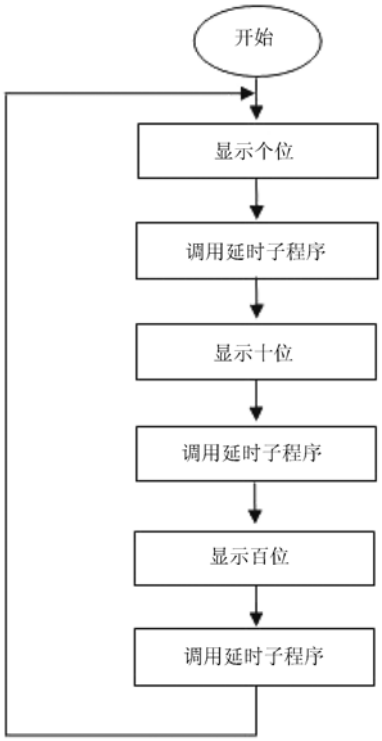


图 13-5　显示子程序流程图

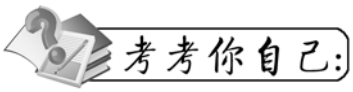
13.6.2　温度计程序清单

```
        ORG 0000H
        AJMP MAIN
        AJMP DISP
        0RG 0003H
        AJMP ADC
MAIN:   SETB IT0            ; INT0 为边沿触发
        MOV IE，#81H        ; 开中断
        MOV SP，#60H        ; 设置堆栈指针
        MOV 30H，#00H       ; 个位显示缓冲区清零
        MOV 31H，#00H       ; 十位显示缓冲区清零
        MOV 32H，#00H       ; 百位显示缓冲区清零
        MOV DPTR，#7FF8H    ; 指向 ADC0809 通道 0 地址
        MOVX @DPTR，A       ; 首次启动 A/D 转换器
```

```
LOOP:   CALL DISP           ；调用显示子程序
        AJMP LOOP           ；循环显示
DISP:                       ；显示子程序
        SETB P2.0           ；关闭百位
        SETB P2.1           ；关闭十位
        CLR P2.2            ；点亮个位
        MOV P1，30H         ；显示个位
        CALL DELAY          ；调用延时子程序
        SETB P2.0           ；关闭百位
        SETB P2.2           ；关闭个位
        CLR P2.1            ；点亮十位
        MOV P1，31H         ；显示十位
        CALL DELAY          ；调用延时子程序
        SETB P2.1           ；关闭十位
        SETB P2.2           ；关闭个位
        CLR P2.0            ；点亮百位
        MOV P1，31H         ；显示百位
        CALL DELAY          ；调用延时子程序
        RET
ADC:                        ；A/D 转换子程序
        MOV DPTR，#7FF8H    ；指向 ADC0809 通道 0 地址
        MOVX A ，@DPTR      ；读取 A/D 转换结果
        MOVX @DPTR，A       ；再次启动 A/D
        MOV DPTR，#TABL1    ；指向温度表首址
        MOVC A，@A+DPTR     ；读取温度值
        MOV B，#64H         ；求出百位数
        DIV AB
        MOV DPTR，#TABL2    ；指向显示码首址
        MOVC A，@A+DPTR     ；查表，求出对应显示码
        MOV 32H，A          ；百位显示码存放 32H 显示缓冲区
        MOV A，B            ；余数送给 A
        MOV A，#0AH         ；求出十位数
```

```
        DIV AB
        MOVC A，@A+DPTR      ；查表，求出对应显示码
        MOV 31H，A           ；十位显示码存放 31H 显示缓冲区
        MOV A，B             ；余数送给 A
        MOVC A，@A+DPTR      ；查表，求出对应显示码
        MOV 30H，A           ；个位显示码存放 30H 显示缓冲区
        RETI
TABL1：                      ；温度表
        DB 00H、00H、00H、00H、00H、00H、00H、00H
        DB 00H、00H、00H、00H、00H、00H、00H、00H
        DB 00H、00H、00H、00H、00H、00H、00H、00H
        DB 00H、00H、01H、01H、02H、02H、03H、03H
        DB 04H、04H、05H、05H、06H、06H、07H、07H
        DB 08H、08H、09H、09H、0AH、0AH、0BH、0BH
        DB 0CH、0CH、0DH、0DH、0EH、0EH、0FH、0FH
        DB 10H、10H、11H、11H、12H、12H、13H、13H
        DB 14H、14H、15H、15H、16H、16H、17H、17H
        DB 18H、18H、19H、19H、1AH、1AH、1BH、1BH
        DB 1CH、1CH、1DH、1DH、1EH、1EH、1FH、1FH
        DB 20H、20H、21H、21H、22H、22H、23H、23H
        DB 24H、24H、25H、25H、26H、26H、27H、27H
        DB 28H、28H、29H、29H、2AH、2AH、2BH、2BH
        DB 2CH、2CH、2DH、2DH、2EH、2EH、2FH、2FH
        DB 30H、30H、31H、31H、32H、32H、33H、33H
        DB 34H、34H、35H、35H、36H、36H、37H、37H
        DB 38H、38H、39H、39H、3AH、3AH、3BH、3BH
        DB 3CH、3CH、3DH、3DH、3EH、3EH、3FH、3FH
        DB 40H、40H、41H、41H、42H、42H、43H、43H
        DB 44H、44H、45H、45H、46H、46H、47H、47H
        DB 48H、48H、49H、49H、4AH、4AH、4BH、4BH
        DB 4CH、4CH、4DH、4DH、4EH、4EH、4FH、4FH
        DB 50H、50H、51H、51H、52H、52H、53H、53H
```

```
        DB 54H、54H、55H、55H、56H、56H、57H、57H
        DB 58H、58H、59H、59H、5AH、5AH、5BH、5BH
        DB 5CH、5CH、5DH、5DH、5EH、5EH、5FH、5FH
        DB 60H、60H、61H、61H、62H、62H、63H、63H
        DB 64H、64H、65H、65H、66H、66H、67H、67H
        DB 68H、68H、69H、69H、6AH、6AH、6BH、6BH
        DB 6CH、6CH、6DH、6DH、6EH、6EH、6FH、6FH
TABL2：                          ；共阳数码管段码表
        DB 0CH、0F9H、0A4H、0B0H、99H、92H、82H、0F8H、80H、90H
```



1. 请简述单片机应用系统的设计步骤。
2. 如何提高系统的抗干扰能力?
3. 运算放大器 LM324 有何作用？
4. 若温度计采用 $\overline{\text{INT1}}$ 中断读取 A/D 转换结果，应如何修改程序。
5. 若采用共阴数码管，如何修改程序。

项目十四　步进电机控制系统

——单片机入门知识

教学目的

掌握：步进电机的基本使用；单片机与步进电机之间的接口电路。
理解：步进电机控制程序的编写。
了解：步进电机的工作原理。

愿你知多点：

在我们的日常生活中，经常使用打印机、扫描仪等办公设备，其运动部件都是步进电机。在这一章中，我们将通过完成步进电机控制系统这一任务，学习单片机在步进电机控制领域的相关知识。

14.1　能力培养

本章通过完成步进电机控制系统这一任务，可以培养以下能力。

1. 能识别步进电机的引脚。
2. 能正确使用步进电机。
3. 能制作步进电机控制系统。

14.2　任务分析

要完成此项任务，需要掌握以下 5 方面知识。

1. 如何使用步进电机。
2. 如何控制步进电机。
3. 如何设计单片机与步进电机之间的接口电路。
4. 如何编写步进电机的控制程序。

14.3 如何使用步进电机

14.3.1 步进电机工作原理

步进电机是一种将脉冲信号转化为相应角位移的执行结构。当步进电机接收到一个脉冲信号后，这个信号就会驱动步进电机按设定的方向转动一个固定的角度，这个固定的角度通常被称为“步距角”，步进电机的旋转是以固定的角度一步一步运行的。这样就可以通过控制脉冲信号的个数来控制角位移量，实现准确定位，同时可以通过控制脉冲信号的频率来控制电机转动速度，实现转速调整。因为步进电机具有上述特性，所以在工业控制领域得到了广泛应用。

步进电机的结构与一般电机相类似，由托架、外壳、转子与定子组成。与一般电机不同的是步进电机的转子与定子有许多细小的齿，转子为磁性材料制成，定子上绕有线圈。根据线圈的配置，步进电机可分为 2 相、4 相和 5 相等。本节中采用的是 2 相步进电机，2 相步进电机有两组线圈，A、C 为一组，B、D 为一组，其结构如图 14-1 所示。

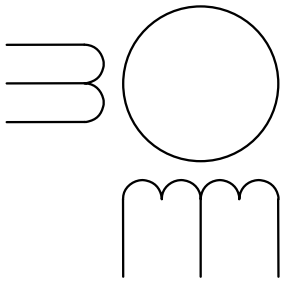


图 14-1　2 相步进电机的结构

14.3.2 步进电机的特点

步进电机具有以下特点。

第一，步进电机不能直接使用交流电源和直流电源供电，须通过脉冲电源供电才能运行。

第二，步进电机的角位移与输入脉冲信号数严格成正比，输出一个脉冲信号，转一个步距角。没有累计误差，具有良好的跟随性。

第三，通过调整控制脉冲信号的频率可以调整电机转速。

第四，通过改变控制脉冲信号的顺序可以改变转动方向。

第五，步进电机的动态响应快，易启动、停止、转向及变速。

第六，步进电机的速度可在相当宽的范围内平稳调节，低速下仍能保证获得较大转矩，因此，通常不用减速器。

14.3.3　步进电机的常数

1.　步进电机的齿距角

齿距角表示步进电机转子或定子相邻两齿中心线之间的夹角，表达式为

$$\theta = 2\pi / Z$$

其中，Z 表示步进电机转子或定子的齿数。

例如对于常用的 2 相 50 齿步进电机，则齿距角为 θ =2π/50 = 7.2°。

2.　步进电机的步距角

步距角表示步进电机每接收到一个脉冲控制信号，转子每走一步所转过的角度，表达式为

$$\varphi = 2\pi / NZ$$

其中，N 表示步进电机的工作拍数；Z 表示步进电机转子或定子的齿数。

例如对于常用的 2 相 50 齿步进电机，其齿距角为 7.2°，转过一个齿距角需要 4 步，这样每步走 1.8°步距角为 1.8°。

14.3.4　步进电机的分类

步进电机按照转子磁性材料的不同可分为反应式、永磁式和混合式步进电机三种。

1.　反应式步进电机（Reaction Stepper Mortor）

反应式步进电机的转子由软磁材料制成，利用齿槽产生磁场。且其结构简单，成本低，步距角可做到很小，但功耗较大，效率较低。

2.　永磁式步进电机（Permanent Magnet Stepper Mortor）

永磁式步进电机的转子由永磁材料制成，转子也是一个磁源。永磁式步进电机输出转矩大，动态性能好，转子与定子具有相同极数，步距角较大。

3.　混合式步进电机（Hybrid Stepiry Mortor）

混合式步进电机结合了反应式和永磁式两种步进电机的优点，其输出转矩大，步距角小，动态性能较好，但结构复杂，成本高。

在上述 3 种步进电机中，由于反应式步进电机的信价比高，所以这种步进电机得到了广泛应用。

14.4 如何控制步进电机

步进电机是由输入脉冲信号进行控制的电机，它的转动角度由输入脉冲信号的数量决定，转动速度由输入脉冲信号的频率所决定。由于步进电机具有上述特点，所以非常适合利用单片机进行控制，下面介绍控制步进电机的原理。

1. 通电换相顺序的控制

步进电机的通电换相顺序是由步进电机的工作方式决定的，通电换相顺序这一过程称为脉冲分配。例如 2 相步进电机的四拍工作方式，若通电顺序 A–B–C–D 为正转，那么通电顺序 D–C–B–A 为反转，所以在使用单片机控制步进电机时，输出控制信号可按照上述顺序来控制各相的通断。

2. 步进电机转向的控制

如果步进电机按照设定的工作方式输入脉冲信号，步进电机正转工作，反相输入脉冲信号步进电机反转工作，所以可以改变脉冲信号的时序来改变步进电机的转向。

3. 步进电机速度的控制

单片机控制系统向步进电机输出一个脉冲信号，步进电机就按照设定的方向转动一个角度。如果单片机输出脉冲信号的间隔越短，步进电机转速就越快，由此可见，改变单片机输出信号的脉冲频率，可以调整步进电机转动的速度。

14.5 如何设计单片机与步进电机之间的接口电路

步进电机接口电路由步进电机、驱动电路、显示电路、按键电路和单片机系统组成，其方框图如图 14-2 所示。图中单片机是整个接口电路的控制中心，其输出的控制信号经驱动电路放大后送到步进电机，使步进电机按要求转动。显示电路主要用于显示步进电机的转速，按键电路主要用于控制步进电机的方向、速度等。

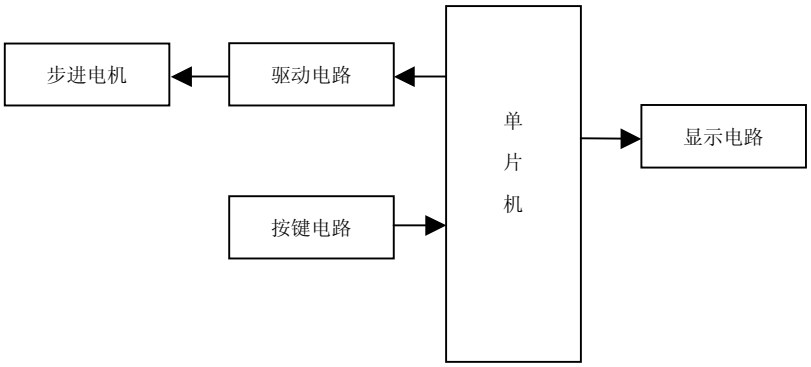


图 14-2　步进电机接口电路方框图

14.5.1　驱动电路

由于步进电机的工作电流较大，而单片机 I/O 端口输出的电流较小，不能直接与步进电机相连接，因此必须在步进电机与单片机之间增加驱动电路。步进电机接口电路中常用的驱动电路有三极管驱动电路和专用集成电路驱动电路两种。

1.　三极管驱动电路

三极管驱动电路如图 14-3 所示。

图中所示为两组结构完全相同的三极管驱动电路，现以左图为例，介绍其工作原理。

当 P1.0=0、P1.1 =1 时，VT3、VT4、VT1 导通，VT5、VT6、VT2 截止。

当 P1.0=1、P1.1 =0 时，VT5、VT6、VT2 导通，VT3、VT4、VT1 截止。

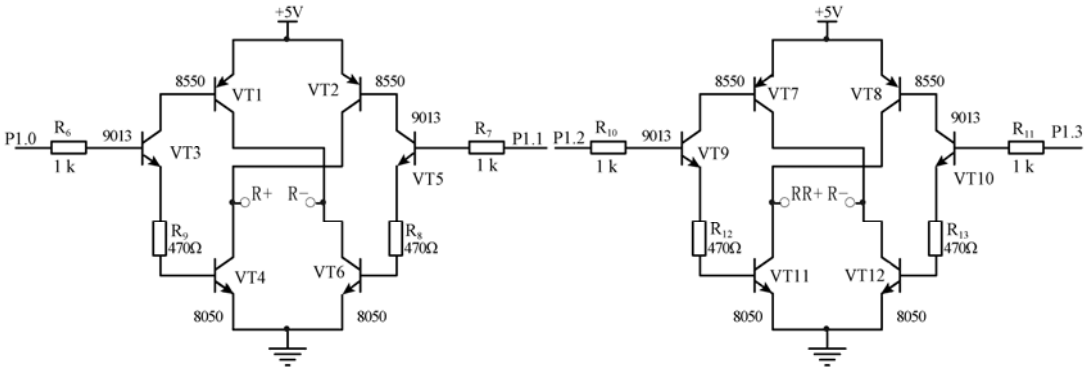


图 14-3　三极管驱动电路

2.　专用集成电路驱动电路

对于电流小于 0.5 A 的小型步进电机，可以采用三极管做驱动电路，对于大电流步进电机，一般采用驱动集成电路。例如 FT5754 型步进电机的专用驱动电路，它是一个 12 引脚的功率集成电路，这种专用驱动电路的驱动电流可以达到 3 A，而且电路在使用过程中较简单。图 14-4 所示为 FT5754 型步进电机的专用集成电路驱动电路。

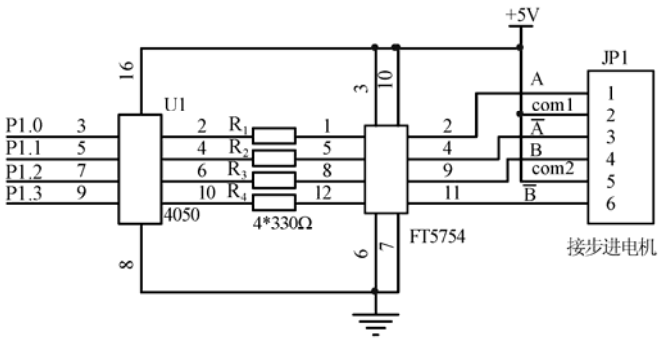


图 14-4　FT5754 型步进电机的专用集成电路驱动电路

14.5.2 显示电路

显示电路如图 14-5 所示。图中所示为 DS1、DS2 和 DS3 为三个共阳数码管，VT13、VT14 和 VT15 为驱动三极管，R_3、R_4 和 R_5 为限流电阻器。单片机 P0 输出字形码（即转速信息）送到数码管的字形端 a～g，单片机 P3.5、P3.6 和 P3.7 输出字位码，经 R_3、R_4 和 R_5 后分别送到三极管 VT13、VT14 和 VT15 的基极。

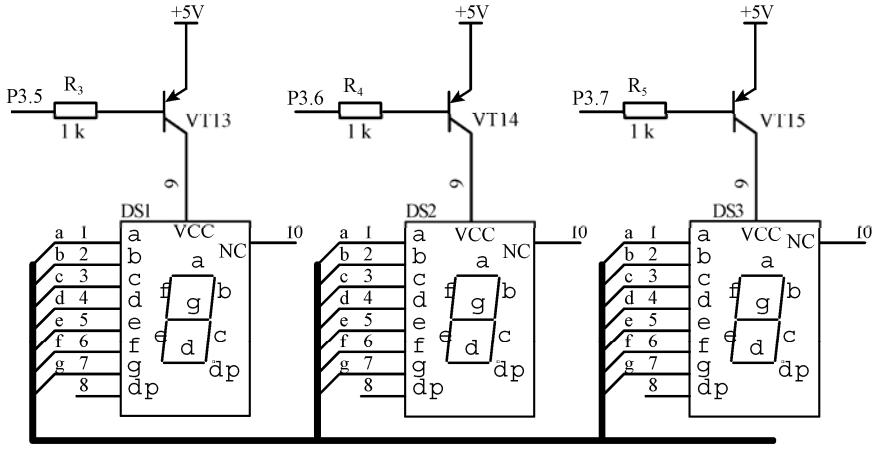


图 14-5　步进电机转速显示电路

14.5.3 步进电机接口电路的整机电路

步进电机接口电路的整机电路如图 14-6 所示。其中 U2、DS1～DS3 组成显示电路，VT1～VT12 为步进电机的驱动电路，W1～W6 为按键电路，主要用于控制步进电机的方向、速度等。

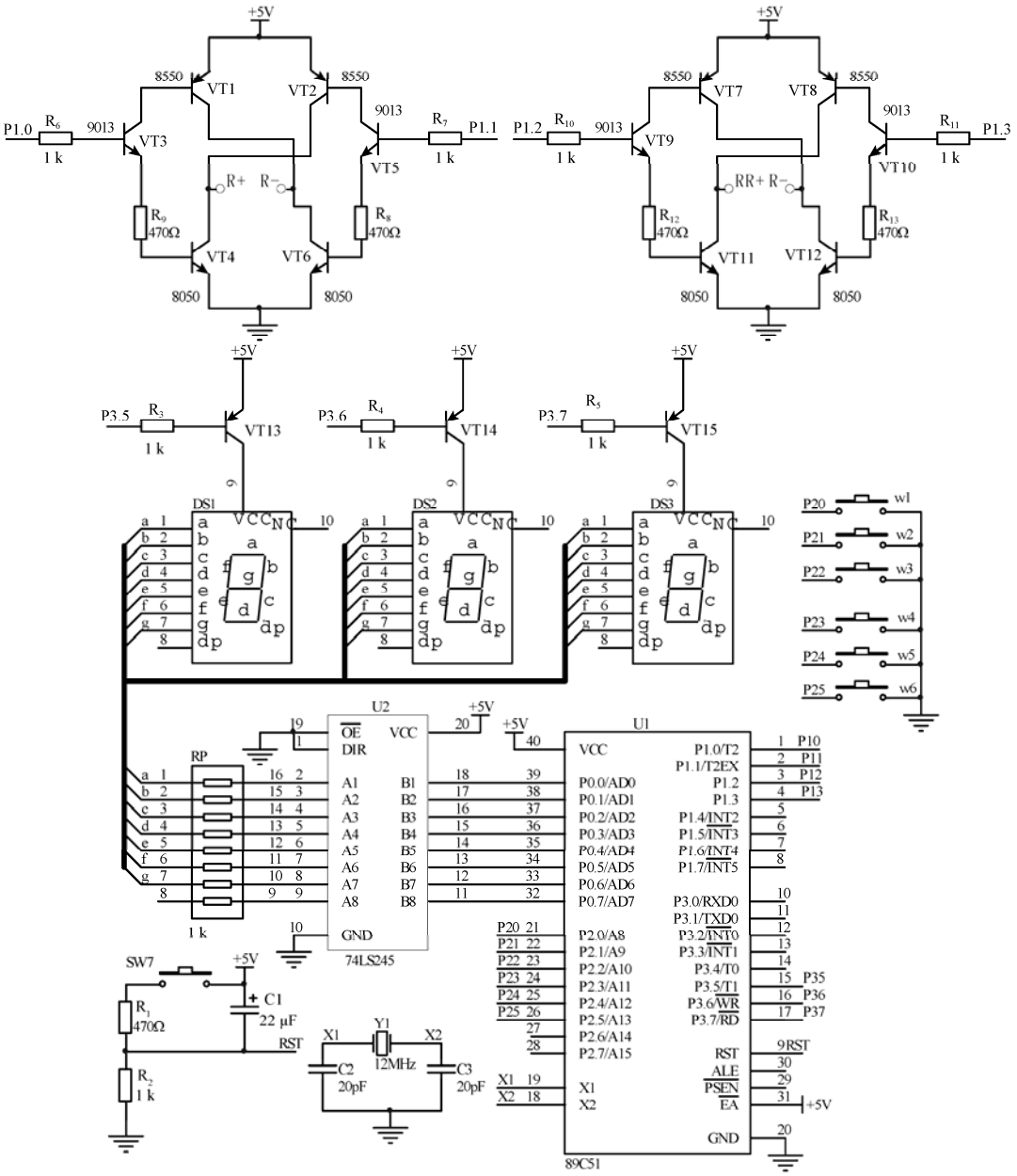


图 14-6　步进电机接口电路的整机电路

14.6　如何编写步进电机的控制程序

14.6.1　步进电机正转控制程序

我们知道步进电机的转向由单片机输出脉冲信号的时序决定，所以要实现本系统的 2

相步进电机正转，在 1 相励磁法方式下，输出正向信号为 0AH→06H→05H→09H，循环 5 次 20 步。步进电机正转程序流程图如图 14-7 所示。

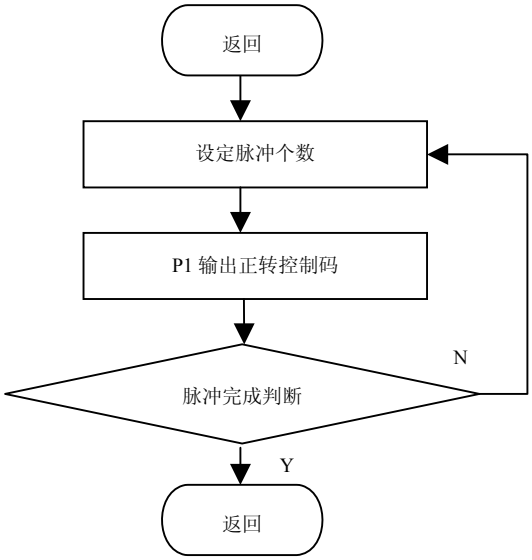


图 14-7　步进电机正转程序流程图

步进电机的正转程序如下所示。

```
        MOV R1,#5       ；用来循环 5 次，完成 1 圈
S1:     MOV P1,#0AH     ；让步进电机正转动 18 度
        CALL DELAY      ；间隔脉冲宽度
        MOV P1,#06H     ；让步进电机正转动 18 度
        CALL DELAY      ；间隔脉冲宽度
        MOV P1,#05H     ；让步进电机正转动 18 度
        CALL DELAY      ；间隔脉冲宽度
        MOV P1,#09H     ；让步进电机正转动 18 度
        CALL DELAY      ；间隔脉冲宽度
        DJNZ R1, S1     ；判断是否为 0
        LJMP R
        RET
```

14.6.2　步进电机反转控制程序

单片机输出反向信号为 09H→05H→06H→0AH，循环 5 次共 20 步。其程序流程图如图 14-8 所示。

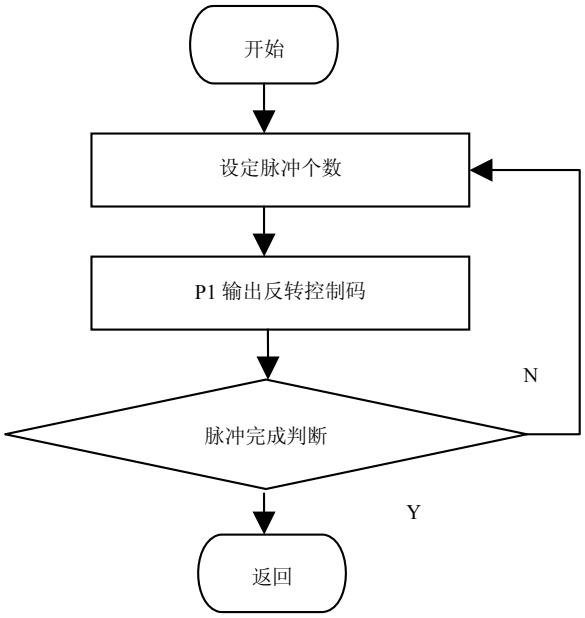


图 14-8　步进电机反程序转流程图

步进电机反转参考程序如下所示。

```
MOV R1,#5            ；用来循环 5 次，完成反转 1 圈
S2: MOV P1,#09H      ；让步进电机反转动 18 度
CALL DELAY           ；间隔脉冲宽度
MOV P1,#05H          ；让步进电机反转动 18 度
CALL DELAY           ；间隔脉冲宽度
MOV P1,#06H;         ；让步进电机反转动 18 度
CALL DELAY           ；间隔脉冲宽度
MOV P1,#0AH          ；让步进电机反转动 18 度
CALL DELA            ；判断是否为 0
```

14.6.3　步进电机加速控制程序

要实现步进电机的加速，可以调整脉冲信号的时间间隔。假设每步间隔时间为 1 s，即转 1 圈的时间为 20 s，那么把间隔时间缩短，就可实现步进电机的加速。步进电机加速程序流程图如图 14-9 所示。

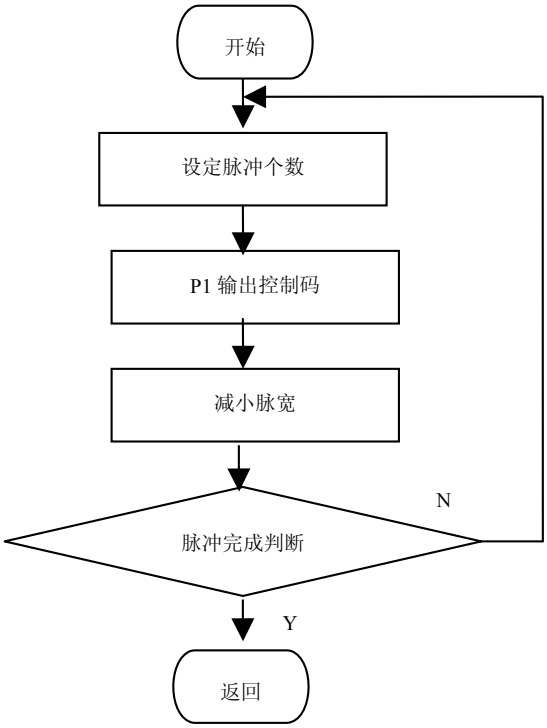


图 14-9　步进电机加速程序流程图

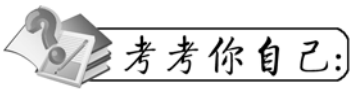
步进电机加速控制参考程序如下所示。

```
Speed:
MOV R7,#50          ；加速
RET
STOP:               ；暂停
CALL XIC
JNB ACC.5,KEYHUI
JMP STOP
KEYHUI:
RET
```

14.6.4　步进电机转速显示程序

要显示步机电机的转速，需要将读取的转换结果转换成二位BCD码，依此按设定时间，将个位与十位显示缓冲送到相应的数码管。步进电机转速显示程序如下所示。

```
XIC:    MOV PSW,#08H
        MOV R1,#100
        MOV R2,#10
        Data: MOV A,R0
        MOV B,R1
        DIV AB
        MOV R3,A        ；得出百位__R3
        MOV A,B
        MOV B,R2
        DIV AB
        MOV R4,A        ；得出十位
        MOV A,B
        MOV B,#1
        DIV AB
        MOV R5,A        ；得出个位__R5
        SETB P3.7       ；个位
        SETB P3.6；
        CLR P3.5；
        MOV A,R3        ；把百位值赋给 A
        MOV DPTR,#TABLE1
        MOVC A,@A+DPTR  ；读码段
        MOV P0,A        ；显示百位
        CALL DELAY5
        SETB P3.7
        CLR P3.6
        SETB P3.5
        MOV A,R4        ；把十位值赋给 A
        MOV DPTR,#TABLE1
        MOVC A,@A+DPTR
        MOV P0,A
        CALL DELAY5
        CLR P3.7
        SETB P3.6
        SETB P3.5
        MOV A,R5
        MOV DPTR,#TABLE1
        MOVC A,@A+DPTR
```



1. 简述步进电机的基本结构。
2. 简述步进电机的控制原理。
3. 如何控制步进电机的方向。
4. 如何控制步进电机的转速。

附录 A　MCS—51 系列单片机指令表

1.　数据传送指令

指令分类	助 记 符	功能说明	字 节 数	机器周期数
以 A 为目的操作数指令	MOV A，Rn	A←（Rn）	1	1
	MOV A，direct	A←（direct）	2	1
	MOV A，#data	A←data	2	1
	MOV A，@Ri	A←（(Ri)）	1	1
以 Rn 为目的操作数指令	MOV Rn，direct	Rn←（direct）	2	2
	MOV Rn，#data	Rn←data	2	1
	MOV Rn，A	Rn←（A）	1	1
以直接地址为目的操作数指令	MOV direct，Rn	direct←（Rn）	2	2
	MOV direct，A	direct←（A）	2	1
	MOV direct，@Ri	direct←（(Ri)）	2	2
	MOV direct，#data	direct←data	3	2
	MOV direct1，direct2	direct1←（direct2）	3	2
以寄存器间接地址为目的操作数指令	MOV @Ri，A	（Ri）←（A）	1	1
	MOV @Ri，direct	（Ri）←（direct）	2	2
	MOV @Ri，#data	（Ri）←data	2	1
16 位数据传送指令	MOV DPTR，#data16	DPTR←data16	3	2
堆栈指令	PUSH direct	sp←(sp)+1，(sp)←(direct)	2	2
	POP direct	direct←((sp))，sp←(sp)-1	2	2
数据交换指令	XCH A，Rn	A↔R_n	1	1
	XCH A，direct	A↔（direct）	2	1
	XCH A，@Ri	A↔（Ri）	1	1
	XCHD A，@Ri	A ↔（Ri）	1	1
	SWAP A	$A_{7\sim4}$↔$A_{3\sim0}$	1	1
外部 RAM 数据传送指令	MOVX A，@DPTR	A←（(DPTR)）	1	2
	MOVX @DPTR，A	（DPTR）←（A）	1	2
	MOVX A，@Ri	A←（(Ri)）	1	2
	MOVX @Ri，A	（Ri）←A。	1	2
查表指令	MOVC A，@A+DPTR	A←((A)+(DPTR))	1	2
	MOVC A，@A+PC	PC←(PC)+1，A←((A)+(PC))	1	2

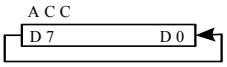
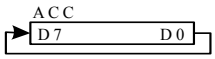
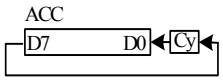
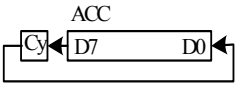
2. 算术运算指令

指令分类	助 记 符	功能说明	字 节 数	机器周期数
不带进位加法指令	ADD A，Rn	A←A+Rn	1	1
	ADD A，direct	A←A+(direct)	2	1
	ADD A，#data	A←A+data	2	1
	ADD A,@Ri	A←A+(Ri)	1	1
带进位加法指令	ADDC A，Rn	A←A+Rn+Cy	1	1
	ADDC A，direct	A←A+(direct) +Cy	2	1
	ADDC A，#data	A←A+data+Cy	2	1
	ADDC A，@Ri	A←A+(Ri) +Cy	1	1
带借位减法指令	SUBB A，Rn	A←A-Rn- Cy	1	1
	SUBB A，direct	A←A-direct- Cy	2	1
	SUBB A，,#data	A←A-data-Cy	2	1
	SUBB A，@Ri	A←A-(Ri)- Cy	1	1
加 1 指令	INC A	A←A+1	1	1
	INC Rn	Rn←Rn+ 1	1	1
	INC direct	(direct) ← (direct)+1	2	1
	INC @Ri	(Ri) ←(Ri)+1	1	1
	INC DPTR	DPTR ←DPTR+1	1	2
减 1 指令	DEC A	A←A-1	1	1
	DEC Rn	Rn←Rn- 1	1	1
	DEC direct	(direct) ← (direct)-1	2	1
	DEC @Ri	(Ri)←(Ri)-1	1	1
乘法指令	MUL AB	BA←A×B	1	4
除法指令	DIV AB	A(商)，B（余数）←A÷B	1	4
十进制数调整指令	DA A	对 A 的内容进行 BCD 码调整	1	1

3. 逻辑运算指令

指令分类	助 记 符	功能说明	字 节 数	机器周期数
逻辑与运算指令	ANL A, Rn	A←A ∧Rn	1	1
	ANL A, direct	A←A ∧（direct）	2	1
	ANL A, @Ri	A←A ∧（Ri）	1	1
	ANL A, #data	A←A ∧data	2	1
	ANL direct, A	（direct）←（direct）∧A	2	1
	ANL direct, #data	（direct）←（direct）∧data	3	2
逻辑或运算指令	ORL A, Rn	A←A ∨Rn	1	1
	ORL A, direct	A←A ∨（direct）	2	1
	ORL A, @Ri	A←A ∨（Ri）	1	1
	ORL A, #data	A←A ∨data	2	1
	ORL direct, A	（direct）←（direct）∨A	2	1
	ORL direct, #data	（direct）←（direct）∨data	3	2
逻辑异或运算指令	XRL A, Rn	A←A ⊕Rn	1	1
	XRL A, direct	A←A ⊕（direct）	2	1
	XRL A, @Ri	A←A ⊕（Ri）	1	1
	XRL A, #data	A←A ⊕data	2	1
	XRL direct, A	（direct）←（direct）⊕A	2	1
	XRL direct, #data	（direct）←（direct）⊕data	3	2
清零	CLR A	A←00	1	1
取反	CPL A	A← $\overline{A}$	1	1

4. 循环移位指令

指令分类	助 记 符	功能说明	字 节 数	机器周期数
不进位指令	RL A		1	1
	RR A		1	1
带进位指令	RRC A		1	1
	RLC A		1	1

5. 控制转移指令

指令分类	助 记 符	功能说明	字 节 数	机器周期数
条件转移指令	LJMP addr16	PC←addr16	3	2
	AJMP addr16	$PC_{15\sim11}$ 不变，$PC_{10\sim0}$←$addr_{10\sim0}$	2	2
	SJMP rel	PC←PC+rel	2	2
	JMP @A+DPTR	PC←A+DPTR	1	2
条件转移指令	JZ rel	若 A≠0，则 PC←PC+2； 若 A＝0，则 PC←PC+2+rel	2	2
	JNZ rel	若 A＝0，则 PC←PC+2； 若 A≠0，则 PC←PC+2+rel	2	2
	JC rel	若 Cy=0，则 PC←PC+2； 若 Cy=1，则 PC←PC+2+rel	2	2
	JNC rel	若 Cy=0，则 PC←PC+2＋rel； 若 Cy=1，则 PC←PC+2	2	2
	JB bit，rel	若（bit）＝0，则 PC←PC+3； 若（bit）＝1，则 PC←C+3+rel	3	2
	JNB bit，rel	若（bit）＝0，则 PC←PC+3+rel；若（bit）＝1， 则 PC←PC+3	3	2
	JBC bit，rel	若（bit）＝0，则 PC←PC+3；若（bit）＝1，则（bit）←0 后 PC←PC+3+rel	3	2
比较转移指令	CJNE A，#data，rel	若 A＝data，则 PC←PC+3，Cy←0；若 A＞data，则 PC←PC+3+rel，Cy←0；若 A＜data，则 PC←PC+3+rel，Cy←1	3	2
	CJNE A，direct，rel	若 A＝（direct），则 PC←PC+3，Cy←0；若 A＞（direct），则 PC←PC+3+rel，Cy←0； 若 A＜（direct），则 PC←PC+3+rel，Cy←1	3	2
	CJNE Rn，＃data，rel	若 Rn＝data，则 PC←PC+3，Cy←0；若 Rn＞data 则 PC←PC+3+rel，Cy←0；若 Rn＜data 则 PC←PC+3+rel，Cy←1	3	2
	CJNE @Ri，#data，rel	若（Ri）＝data，则 PC←PC+3，Cy←0；若（Ri）＞data，则 PC←PC+3+rel，Cy←0； 若（Ri）＜data，则 PC←PC+3+rel，Cy←1	3	2

（续表）

指令分类	助 记 符	功能说明	字 节 数	机器周期数
减 1 不等于 0 转移指令	DJNZ　Rn，rel	①Rn←（Rn）-1 ②若 Rn≠0，则 PC←PC+2+rel； 若（Rn）=0，则 PC←PC+2	2	2
	DJNZ　direct，rel	①direct←（direct）-1 ②若(direct)≠0，则 PC←PC+3+rel； 若（direct）=0，则 PC←PC+3	3	2
调用指令	LCALL addr16	PC←PC+3 SP←SP+1 SP←$PC_{7\sim0}$ SP←SP+1 SP←$PC_{15\sim8}$ PC←addr16	3	2
	ACALL addr11	PC←PC+2 SP←SP+1 SP←$PC_{7\sim0}$ SP←SP+1 SP←$PC_{15\sim8}$ $PC_{10\sim0}$←addr11	2	2
子程序返回指令	RET	$PC_{15\sim8}$←(SP) SP←SP-1 $PC_{7\sim0}$←SP SP←SP-1	1	2
中断返回指令	RETI	$PC_{15\sim8}$←(SP) SP←SP-1 $PC_{7\sim0}$←（SP） SP←SP-1	1	2

6. 位指令

指令分类	助 记 符	功能说明	字 节 数	机器周期数
位传送指令	MOV C，bit	Cy←bit	2	1
	MOV bit，C	bit←Cy	2	2
位清零指令	CLR C	Cy←0	1	1
	CLR bit	bit←0	2	1
位置 1 指令	SETB C	Cy←1	1	1
	SETB bit	bit←1	2	1
位操作指令	ANL C，bit	Cy←Cy∧ bit	2	2
	ANL C，/bit	Cy←Cy∧ $\overline{bit}$	2	2
	ORL C，bit	Cy←Cy∨ bit	2	2
	ORL C，/bit	Cy←Cy∨ $\overline{bit}$	2	2
	CPL C	Cy← $\overline{Cy}$	1	1
	CPL bit	bit← $\overline{bit}$	2	1

7. 空操作指令

指令分类	助 记 符	功能说明	字 节 数	机器周期数
空操作指令	NOP	PC←PC+1	1	1

附录 B　ASCII 码表

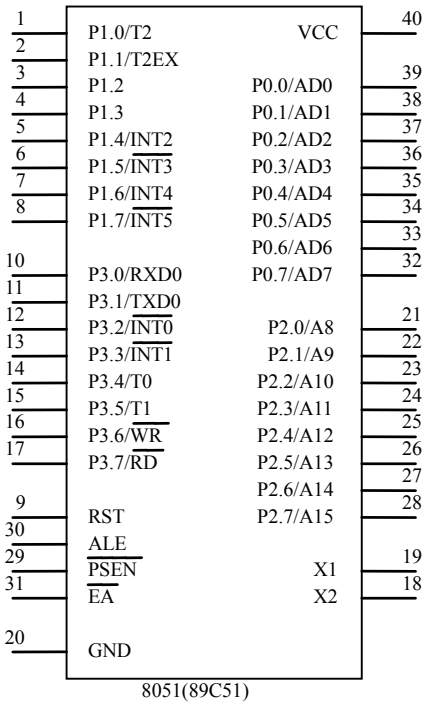
下表列出了 ASCII 字符集。每一个字符有 ASCII 码的十进制值、十六进制值和对应字符。

十进制数	十六进制数	字　符	十进制数	十六进制数	字　符	十进制数	十六进制数	字　符
0	00	NUL	24	18	CAN	48	30	0
1	01	SOH	25	19	EM	49	31	1
2	02	STX	26	1A	SUB	50	32	2
3	03	X	27	1B	ESC	51	33	3
4	04		28	1C	FS	52	34	4
5	05	ENQ	29	1D	GS	53	35	5
6	06	ACK	30	1E	RS	54	36	6
7	07	BEL	31	1F	US	55	37	7
8	08	BS	32	20	SPACE	56	38	8
9	09	TAB	33	21	!	57	39	9
10	0A	LF	34	22	″	58	3A	:
11	0B	VT	35	23	#	59	3B	;
12	0C	FF	36	24	$	60	3C	<
13	0D	CR	37	25	%	61	3D	=
14	0E	SO	38	26	&	62	3E	>
15	0F	SI	39	27	‘	63	3F	?
16	10	DLE	40	28	(	64	40	@
17	11	DC1	41	29	)	65	41	A
18	12	DC2	42	2A	*	66	42	B
19	13	DC3	43	2B	+	67	43	C
20	14	DC4	44	2C	,	68	44	D
21	15	NAK	45	2D	-	69	45	E
22	16	SYN	46	2E	.	70	46	F
23	17	ETB	47	2F	/	71	47	G

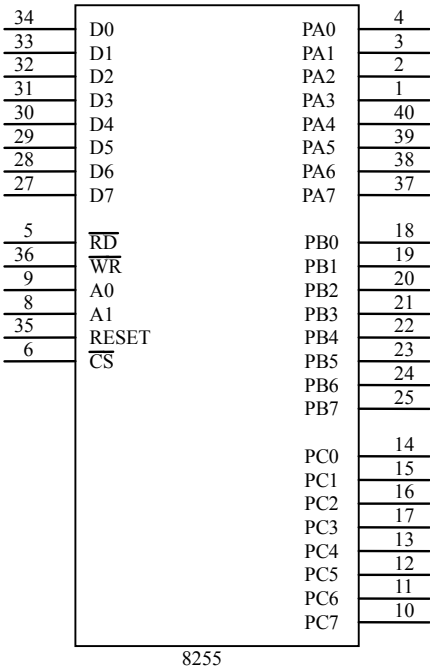
（续表）

十进制数	十六进制数	字　符	十进制数	十六进制数	字　符	十进制数	十六进制数	字　符
72	48	H	91	5B	[	110	6E	n
73	49	I	92	5C	\	111	6F	o
74	4A	J	93	5D	]	112	70	p
75	4B	K	94	5E	^	113	71	q
76	4C	L	95	5F	—	114	72	r
77	4D	M	96	60	`	115	73	s
78	4E	N	97	61	a	116	74	t
79	4F	O	98	62	b	117	75	u
80	50	P	99	63	c	118	76	v
81	51	Q	100	64	d	119	77	w
82	52	R	101	65	e	120	78	x
83	53	S	102	66	f	121	79	y
84	54	T	103	67	g	122	7A	z
85	55	U	104	68	h	123	7B	{
86	56	V	105	69	i	124	7C	|
87	57	W	106	6A	j	125	7D	}
88	58	X	107	6B	k	126	7E	~
89	59	Y	108	6C	l	127	7F	del
90	5A	Z	109	6D	m			

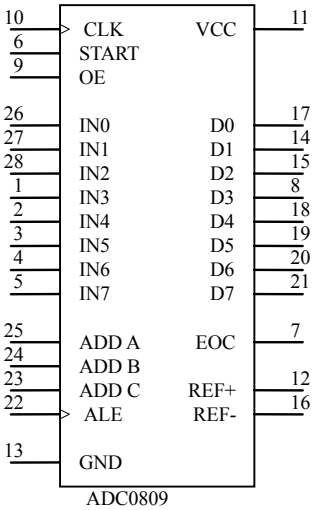
附录 C　常用芯片引脚



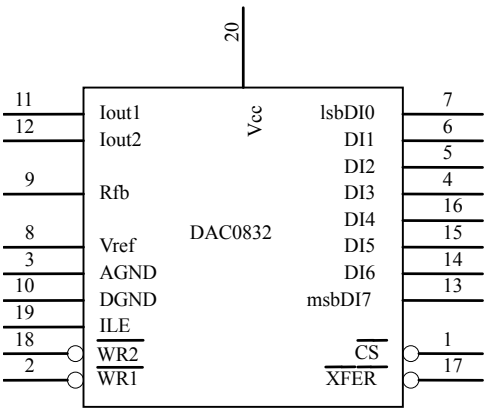
8 位 CPU



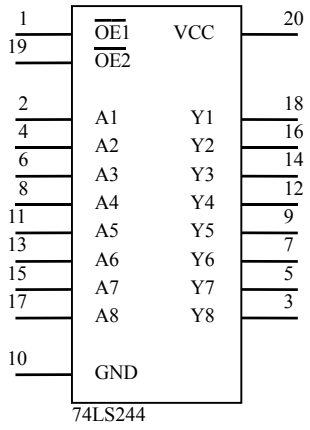
I/O 扩展集成电路



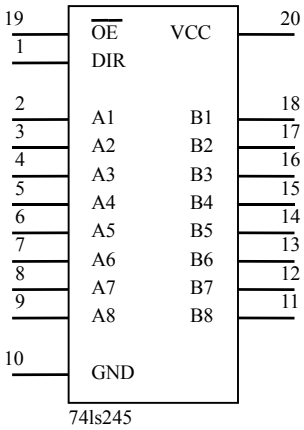
8 位 A/D 转换器



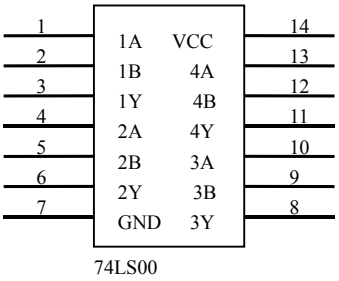
8 位 D/A 转换器



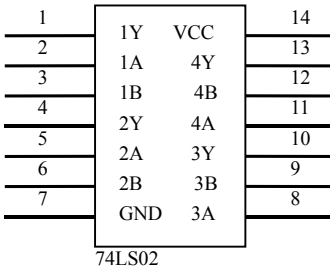
单向驱动器



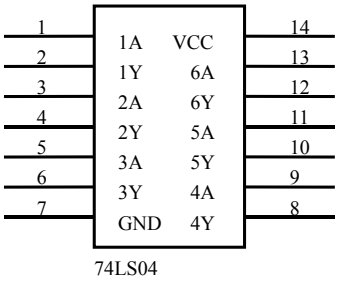
双向驱动器



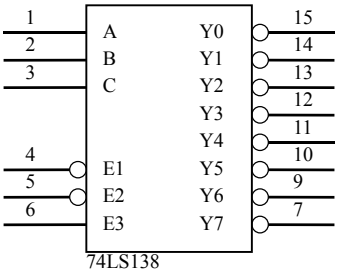
四路二输入与非门



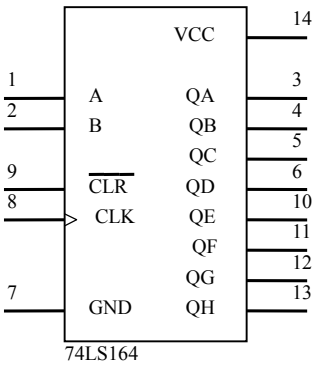
四路二输入或非门



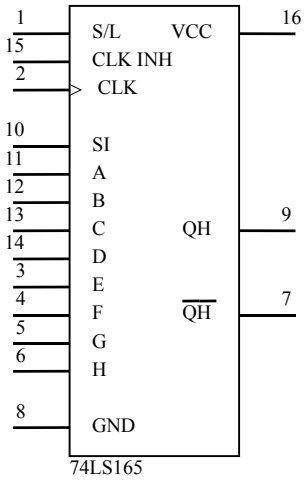
六路反向器



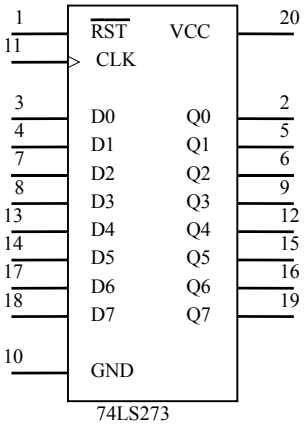
译码器

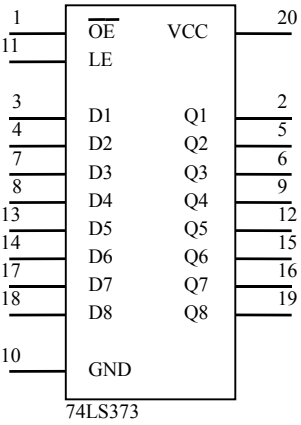


8 位串入/并出移位寄存器

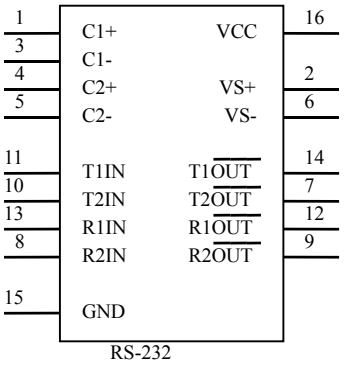


8 位并入/串出移位寄存器

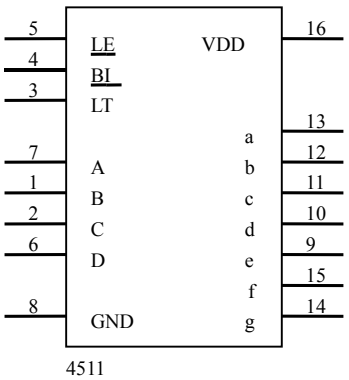




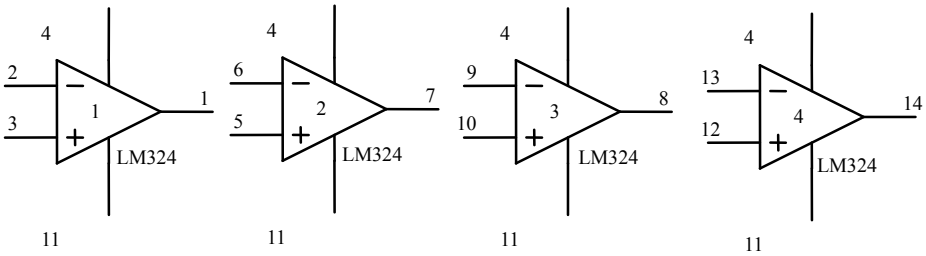
8D 锁存器



BCD 译码器



RS—232 通信芯片


4
2
3
1
1
LM324
11
4
6
5
2
7
LM324
11
4
9
10
3
8
LM324
11
4
13
12
4
14
LM324
11

四路运算放大器

附录 D　单片机装调工专项能力认证鉴定标准（中级）

一、知识要求

1. 掌握单片机的基本结构。
2. 掌握指令系统的基本使用方法（包括数据传送指令、算术指令、逻辑指令、位操作指令、控制及转移指令）。
3. 理解键盘和显示系统的结构与工作原理（包括硬件与软件）。
4. 理解定时器的结构与工作原理（包括硬件与软件）。
5. 理解中断的结构与工作原理（包括硬件与软件）。
6. 掌握常用元器件的结构、符号、测量与基本使用方法（包括电阻器、电容器、二极管、三极管、蜂鸣器等）。
7. 掌握伟福调试软件的基本使用方法。
8. 掌握元器件的整形与焊接。
9. 掌握直流电源的基本使用方法。
10. 掌握万用表的基本使用方法。
11. 掌握编程器的基本使用方法。

二、技能要求

1. 具有硬件系统的安装、调试与维修能力。
2. 具有编写简单的单片机程序的能力。
3. 具有硬件与软件结合的综合调试能力。
4. 具有熟练使用万用表、直流电源的能力。
5. 具有熟练使用伟福仿真软件及仿真机的能力。
6. 具有熟练使用编程器的能力。

三、考核方法

单片机应用技能考核时间为 180 分钟（共三个小时），全部为实际操作的考试，学生允许带一本参考教材。

实际操作的考试要求考生先对照所给出的电路图，焊接电路，然后根据题目的具体要求，编写应用软件，并进行硬件与软件的综合调试，完成整个产品的制作。

1. 焊接与调试电路

考生先用万用表测量元件好坏，然后对照试卷所给出的电路图焊接电路（使用万用板），并进行硬件调试。要求元器件的整形美观、布局合理、焊点光滑、无毛刺、无虚焊、

无漏焊。

2. 编写与调试软件

考生根据试题要求，利用伟福仿真软件编写软件程序，并进行软件调试。考生编好的软件要符合试卷要求，且可读性强，逻辑合理。

3. 硬件与软件的综合调试

将调试好的软件程序通过编程器烧写入芯片，并进行硬件与软件的综合调试，使硬件与软件均符合试卷要求。

四、评分标准

1. 元件整形焊接（40 分）

（1） 要求

① 元器件整形美观、布局合理。

② 焊点光滑、无毛刺、无虚焊、漏焊。

（2）扣分原则

序　号	项　目	总　分	扣 分 原 则
1	元器件整形	5	整形不规范，一个元器件扣 0.5 分。
2	元器件布局	10	根据具体布局情况，酌情扣 0～10 分。
3	元器件焊接	25	漏焊一个元器件扣 1 分；虚焊一个元器件扣 1 分；焊点一处不光滑或有毛刺扣 0.5 分。

2. 软件的编写与调试（50 分）

（1） 要求

根据题目要求，编写结构合理、可读性强的软件程序，并进行调试。

（2） 扣分原则

序　号	项　目	总　分	扣 分 原 则
1	编写软件程序	35	写错一条指令扣 1 分；漏写一条指令扣 1 分。
2	优化软件程序	10	结构不合理扣酌情扣 0～10 分。
3	调试、产生 OBJ 或 HEX 文件	5	产生的 OBJ 或 HEX 文件不正确酌情扣 2～4 分；没有产生 OBJ 或 HEX 文件扣 5 分。

3. 软件的烧写（10 分）

（1） 要求

能将产生的 OBJ 或 HEX 文件正确烧写入芯片。

（2） 扣分原则

序　号	项　目	总　分	扣 分 原 则
1	烧写软件程序	10	没有烧写入扣 10 分；烧写入错误酌情扣 1～10 分。

五、考试的组织

本技能鉴定由劳动厅鉴定中心统一组织，统一收费，其中技能考核元器件费用为 30 元，考生考完后，电路板统一上缴给省鉴定中心。

六、参考题目

1. 交通灯控制
2. 彩灯控制
3. 抢答器
4. LED 灯控制
5. 定时显示
6. 学号显示

单片机装调工专项能力认证试题中级技能试题 1

姓名：_____ 准考证号：_____ 成绩：_____

第一部分　元器件整形焊接（40 分）

一、考试要求

1. 将需要焊接的元器件整形。
2. 图 D-1 所示为霓红灯电路，根据图 D-1 焊接电路。

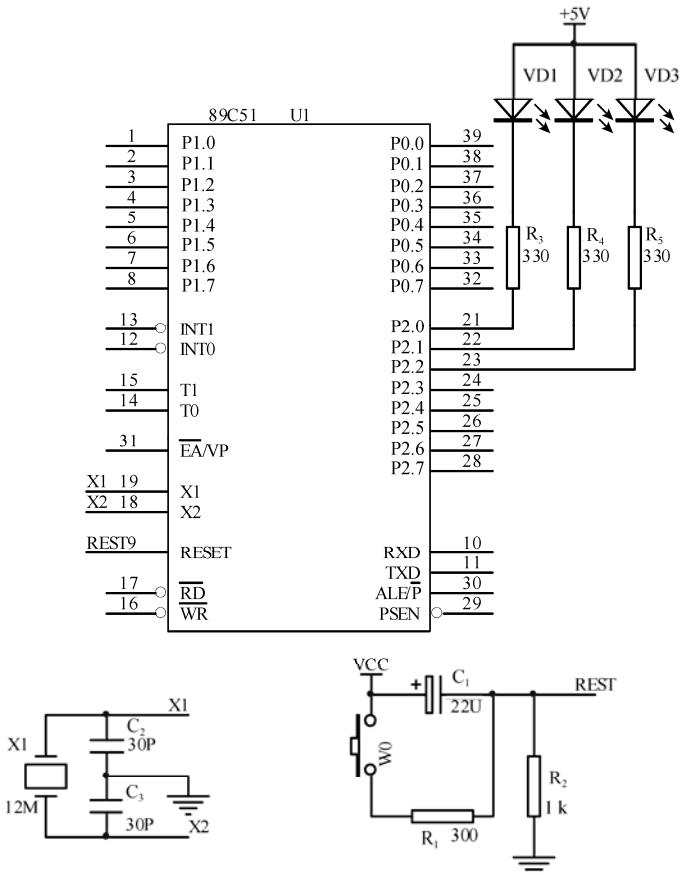


图 D-1　霓虹灯电路

二、评分方法

1. 元器件整形正确、美观得 5 分；整形不规范，一个元器件扣 0.5 分。

2. 元器件布局合理得 10 分；元器件布局不合理扣 1～10 分。

3. 能正确焊接，且焊点光滑美观得 25 分，漏焊一个扣 1 分；虚焊一个扣 0.5 分；焊点一处不光滑或有毛刺扣 0.5 分。元器件整形焊接评分表见下表所列。

元器件整形焊接评分表

项　目	总　分	得　分
元器件整形	5	
元器件布局	10	
元器件焊接	25	

时间：_____　考评员：_____　评分：_____

第二部分　编写、调试软件程序（50 分）

一、考试要求

1. 编写软件程序，要求发光二极管 VD1～VD3 从左至右每隔 0.2 秒循环点亮，电路如图 D-1 所示。

2. 程序结构合理、可读性强。

3. 将原文件转换为 OBJ 或 HEX 文件。

二、评分方法

1. 能正确编写软件程序得 35 分，写错一条指令扣 1 分；漏写一条指令扣 1 分。

2. 软件程序优化合理得 10 分，软件结构不合理扣酌情扣 1～10 分。

3. 能正确调试、产生 OBJ 或 HEX 文件得 5 分，产生的 OBJ 或 HEX 文件不正确酌情扣 2～4 分；没有产生 OBJ 或 HEX 文件扣 5 分。编写、调试软件程序评分表见下表所列。

编写、调试软件程序评分表

项　目	总　分	得　分
编写软件程序	35	
优化软件程序	10	
调试、产生 OBJ 或 HEX 文件	5	

时间：_____　考评员：_____　评分：_____

第三部分　烧写软件（10 分）

一、考试要求

将 OBJ 或 HEX 文件正确烧写入芯片。

二、评分方法

能将产生的 OBJ 或 HEX 文件正确烧写入芯片得 10 分，没有烧写入扣 10 分；烧写入错误酌情扣 1～10 分。软件烧写评分表见下表所列。

软件烧写评分表

项　　目	总　　分	得　　分
烧写软件	10	

时间：_____　考评员：_____　评分：_____

单片机装调工专项能力认证试题中级技能试题 2

姓名：_____ 准考证号：_____ 成绩：_____

第一部分　元器件整形焊接（40 分）

一、考试要求

1. 将需要焊接的元器件整形。
2. 图 D-2 所示为广告灯电路根据图 D-2 焊接电路。

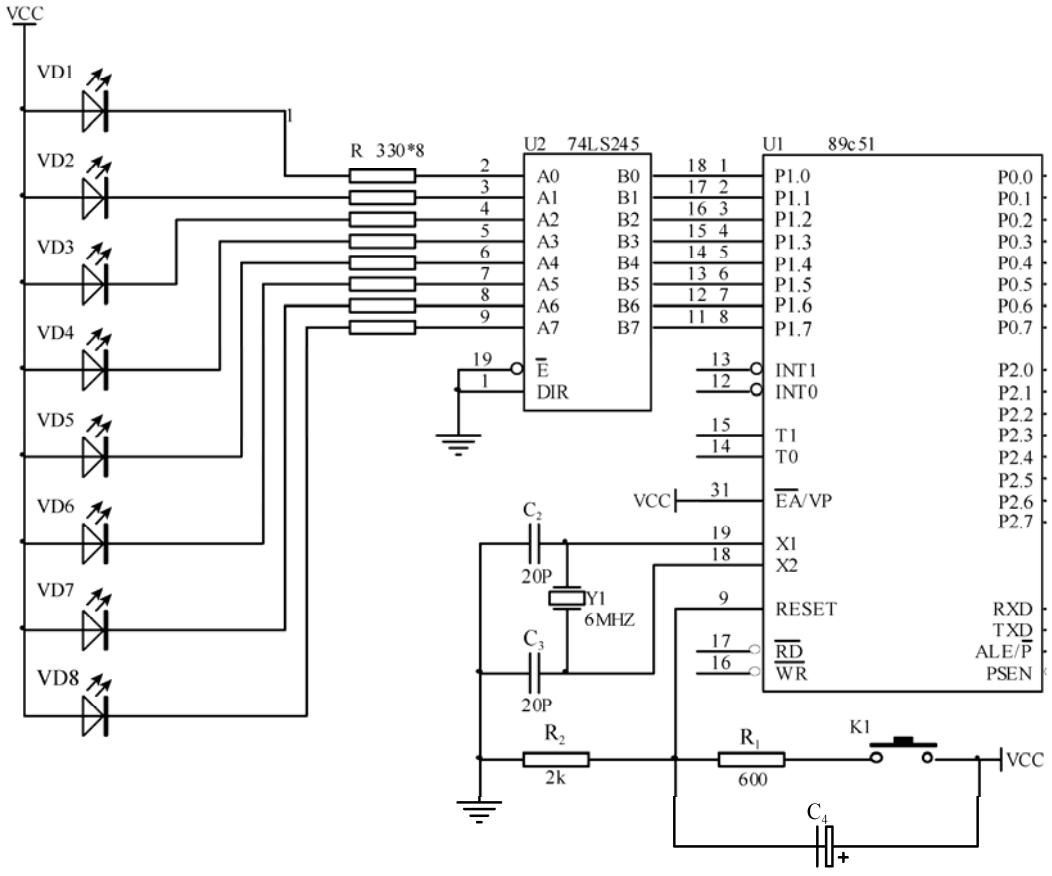


图 D-2　广告灯电路

二、评分方法

1. 元器件整形正确、美观得 5 分；整形不规范，一个元器件扣 0.5 分。

2. 元器件布局合理得 10 分；元件布局不合理扣 1～10 分。

3. 能正确焊接，且焊点光滑美观得 25 分，漏焊一个扣 1 分；虚焊一个扣 0.5 分；焊点一处不光滑或有毛刺扣 0.5 分。元器件整形焊接评分表见下表所列。

元器件整形焊接评分表

项　　目	总　　分	得　　分
元器件整形	5	
元器件布局	10	
元器件焊接	25	

时间：_____　考评员：_____　评分：_____

第二部分　编写、调试软件（50 分）

一、考试要求

1. 编写广告灯软件程序，要求实现发光二极管 VD1～VD8 每隔 0.2 秒从上至下循环点亮，电路如图 D-2 所示。

2. 程序结构合理、可读性强。

3. 将原文件转换为 OBJ 或 HEX 文件。

二、评分方法

1. 能正确编写软件程序得 35 分，写错一条指令扣 1 分；漏写一条指令扣 1 分。

2. 软件优化合理得 10 分，软件结构不合理扣酌情扣 1～10 分。

3. 能正确调试、产生 OBJ 或 HEX 文件得 5 分，产生的 OBJ 或 HEX 文件不正确酌情扣 2～4 分；没有产生 OBJ 或 HEX 文件扣 5 分。编写、调试软件程序评分表见下表所列。

编写、调试软件程序评分表

项　　目	总　　分	得　　分
编写软件程序	35	
优化软件程序	10	
调试、产生 OBJ 或 HEX 文件	5	

时间：_____　考评员：_____　评分：_____

第三部分　烧写软件（10 分）

一、考试要求

将 OBJ 或 HEX 文件正确烧写入芯片。

二、评分方法

能将产生的 OBJ 或 HEX 文件正确烧写入芯片得 10 分，没有写入烧扣 10 分；烧写入错误酌情扣 1～10 分。软件烧写评分表见下表所列。

软件烧写评分表

项　　目	总　　分	得　　分
烧写软件	10	

时间：_____　考评员：_____　评分：_____

单片机装调工专项能力认证试题中级技能试题 3

姓名：_____ 准考证号：_____ 成绩：_____

第一部分　元器件整形焊接（40 分）

一、考试要求

1. 将需要焊接的元器件整形。
2. 图 D-3 所示为单片机抢答控制器根据图 D-3 焊接电路。

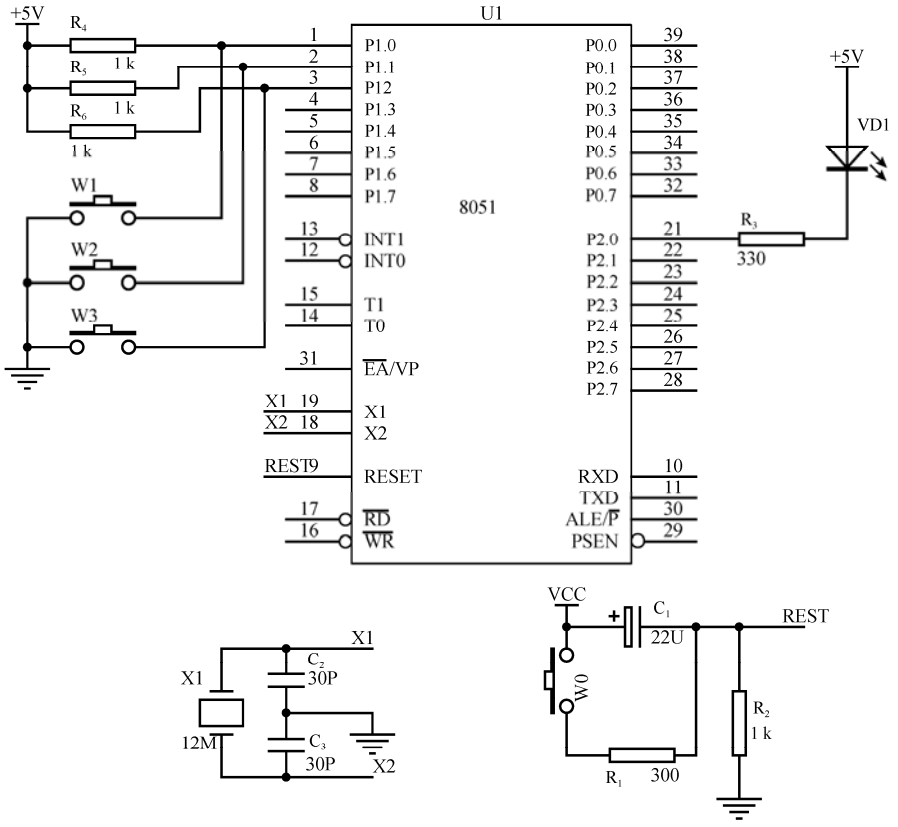


图 D-3　单片机抢答控制器

二、评分方法

1. 元器件整形正确、美观得 5 分；整形不规范，一个元器件扣 0.5 分。

2. 元器件布局合理得 10 分；元器件布局不合理扣 1～10 分。

3. 能正确焊接，且焊点光滑美观得 25 分，漏焊一个扣 1 分；虚焊一个扣 0.5 分；焊点一处不光滑或有毛刺扣 0.5 分。元器件整形焊接评分表见下表所列。

元器件整形焊接评分表

项　目	总　分	得　分
元器件整形	5	
元器件布局	10	
元器件焊接	25	

时间：_____　考评员：_____　评分：_____

第二部分　编写、调试软件程序（50 分）

一、考试要求

1. 抢答器应能实现抢答功能，即先按者有效。
2. 抢答器在有人按下时，灯亮。
3. 抢答器应能在控制者的控制下，再次进行抢答。
4. 程序结构合理、可读性强。
5. 将原文件转换为 OBJ 或 HEX 文件。

二、评分方法

1. 能正确编写软件程序得 35 分，写错一条指令扣 1 分；漏写一条指令扣 1 分。

2. 软件程序优化合理得 10 分，软件程序结构不合理扣酌情扣 1～10 分。

3. 能正确调试、产生 OBJ 或 HEX 文件得 5 分，产生的 OBJ 或 HEX 文件不正确酌情扣 2～4 分；没有产生 OBJ 或 HEX 文件扣 5 分。编写、调试软件程序评分表见下表所列。

编写、调试软件程序评分表

项　目	总　分	得　分
编写软件程序	35	
优化软件程序	10	
调试、产生 OBJ 或 HEX 文件	5	

时间：_____　考评员：_____　评分：_____

第三部分　烧写软件（10 分）

一、考试要求

将 OBJ 或 HEX 文件正确烧写入芯片。

二、评分方法

能将产生的 OBJ 或 HEX 文件正确烧写入芯片得 10 分，没有烧写入扣 10 分；烧写入错误酌情扣 1～10 分。软件烧写评分表见下表所列。

软件烧写评分表

项　　目	总　　分	得　　分
烧写软件	10	

时间：_____　考评员：_____　评分：_____

单片机装调工专项能力认证试题中级技能试题 4

姓名：_____ 准考证号：_____ 成绩：_____

第一部分　元器件整形焊接（40 分）

一、考试要求

1. 将需要焊接的元件整形。
2. 图 D-4 所示为秒表时钟电路，根据图 D-4 焊接电路。

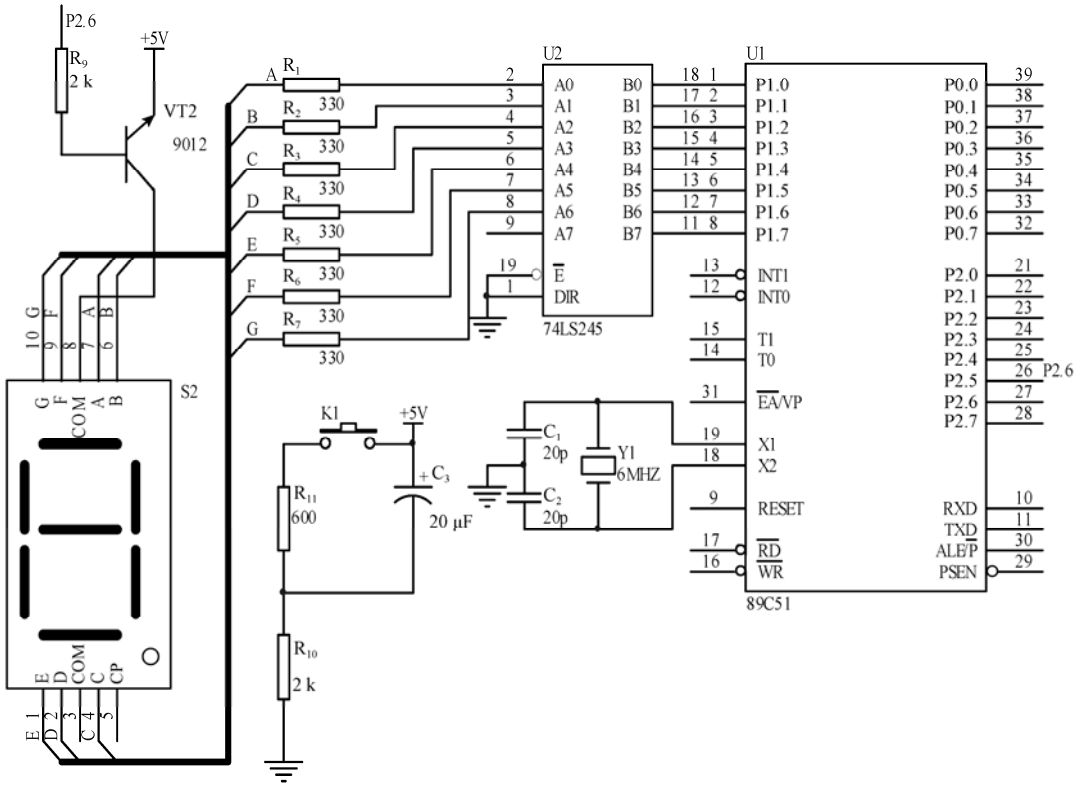


图 D-4　秒表时钟电路

二、评分方法

1. 元器件整形正确、美观得 5 分；整形不规范，一个元件扣 0.5 分。

2. 元器件布局合理得 10 分；元器件布局不合理扣 1～10 分。

3. 能正确焊接，且焊点光滑美观得 25 分，漏焊一个扣 1 分；虚焊一个扣 0.5 分；焊点一处不光滑或有毛刺扣 0.5 分。元器件整形焊接评分表见下表所列。

元器件整形焊接评分表

项　目	总　分	得　分
元器件整形	5	
元器件布局	10	
元器件焊接	25	

时间：_____　考评员：_____　评分：_____

第二部分　编写、调试软件程序（50 分）

一、考试要求

1. 编写软件程序，要求完成秒表计数功能，能从 00～59 范围内加 1 计数，每隔 1 秒进行一次加 1 计数。

2. 程序结构合理、可读性强。

3. 将原文件转换为 OBJ 或 HEX 文件。

二、评分方法

1. 能正确编写软件程序得 35 分，写错一条指令扣 1 分；漏写一条指令扣 1 分。

2. 软件程序优化合理得 10 分，软件程序结构不合理扣酌情扣 1～10 分。

3. 能正确调试、产生 OBJ 或 HEX 文件得 5 分，产生的 OBJ 或 HEX 文件不正确酌情扣 2～4 分；没有产生 OBJ 或 HEX 文件扣 5 分。编写、调试软件程序评分表见下表所列。

编写、调试软件程序评分表

项　目	总　分	得　分
编写软件程序	35	
优化软件程序	10	
调试、产生 OBJ 或 HEX 文件	5	

时间：_____　考评员：_____　评分：_____

第三部分　烧写软件（10 分）

一、考试要求

将 OBJ 或 HEX 文件正确烧写入芯片。

二、评分方法

能将产生的 OBJ 或 HEX 文件正确烧写入芯片得 10 分，没有烧写入扣 10 分；烧写入错误酌情扣 1～10 分。软件烧写评分表见下表所列。

软件烧写评分表

项　　目	总　　分	得　　分
烧写软件	10	

时间：_____　考评员：_____　评分：_____

单片机装调工专项能力认证试题中级技能试题 5

姓名：_____ 准考证号：_____ 成绩：_____

第一部分　元器件整形焊接（40 分）

考试要求

1. 将需要焊接的元器件整形。
2. 图 D-5 所示为交通灯电路，根据图 D-5 焊接电路。

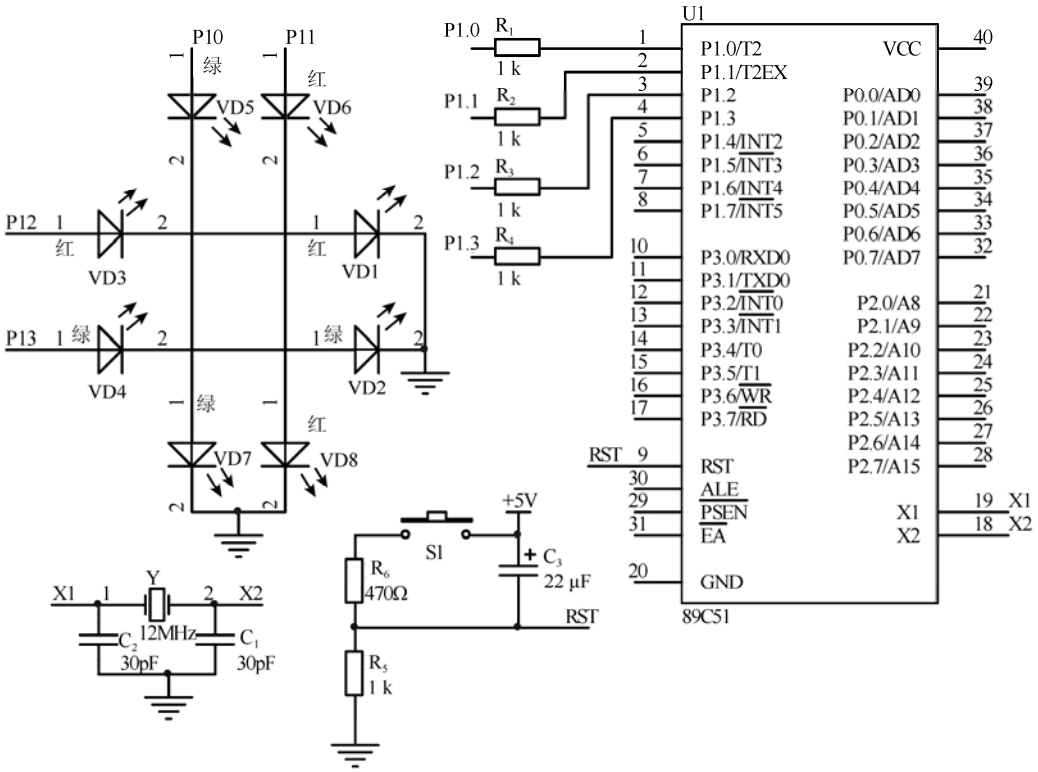


图 D-5　交通灯电路

二、评分方法

1. 元器件整形正确、美观得 5 分；整形不规范，一个元器件扣 0.5 分。

2. 元器件布局合理得 10 分；元件布局不合理扣 1～10 分。

3. 能正确焊接，且焊点光滑美观得 25 分，漏焊一个扣 1 分；虚焊一个扣 0.5 分；焊点一处不光滑或有毛刺扣 0.5 分。元器件整形焊接评分表见下表所列。

元器件整形焊接评分表

项　目	总　分	得　分
元器件整形	5	
元器件布局	10	
元器件焊接	25	

时间：_____　考评员：_____　评分：_____

第二部分　编写、调试软件程序（50 分）

一、考试要求

1. 编写软件程序，要求完成交通灯功能，红绿灯每隔 30 秒轮换点亮一次。

2. 程序结构合理、可读性强。

3. 将原文件转换为 OBJ 或 HEX 文件。

二、评分方法

1. 能正确编写软件程序得 35 分，写错一条指令扣 1 分；漏写一条指令扣 1 分。

2. 软件程序优化合理得 10 分，软件程序结构不合理扣酌情扣 1～10 分。

3. 能正确调试、产生 OBJ 或 HEX 文件得 5 分，产生的 OBJ 或 HEX 文件不正确酌情扣 2～4 分；没有产生 OBJ 或 HEX 文件扣 5 分。编写、调试软件程序评分表见下表所列。

编写、调试软件程序评分表

项　目	总　分	得　分
编写软件程序	35	
优化软件程序	10	
调试、产生 OBJ 或 HEX 文件	5	

时间：_____　考评员：_____　评分：_____

第三部分　烧写软件（10 分）

一、考试要求

将 OBJ 或 HEX 文件正确烧写入芯片。

二、评分方法

能将产生的 OBJ 或 HEX 文件正确烧写入芯片得 10 分，没有烧写入扣 10 分；烧写入错误酌情扣 1～10 分。软件烧写评分表见下表所列。

软件烧写评分表

项　　目	总　　分	得　　分
烧写软件	10	

时间：_____　考评员：_____　评分：_____

单片机装调工专项能力认证试题中级技能试题 6

姓名：_____ 准考证号：_____ 成绩：_____

第一部分　元器件整形焊接（40 分）

一、考试要求

1. 将需要焊接的元器件整形。
2. 图 D-6 所示为独立键盘电路，根据图 D-6 焊接电路。

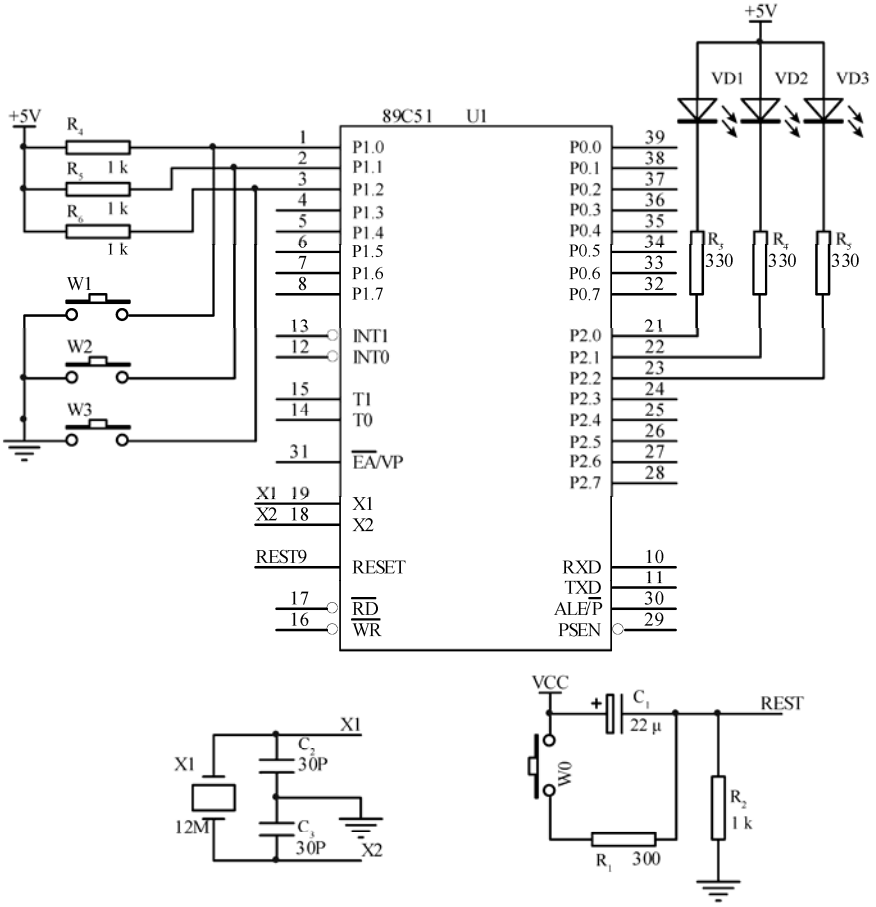


图 D-6　独立键盘电路

二、评分方法

1. 元器件整形正确、美观得 5 分；整形不规范，一个元件扣 0.5 分。

2. 元器件布局合理得 10 分；元器件布局不合理扣 1～10 分。

3. 能正确焊接，且焊点光滑美观得 25 分，漏焊一个扣 1 分；虚焊一个扣 0.5 分；焊点一处不光滑或有毛刺扣 0.5 分。元器件整形焊接评分表见下表所列。

元器件整形焊接评分表

项　　目	总　　分	得　　分
元器件整形	5	
元器件布局	10	
元器件焊接	25	

时间：_____　考评员：_____　评分：_____

第二部分　编写、调试软件程序（50 分）

一、考试要求

1. 编写软件，要求按下按钮，对应发光二极管点亮，如按下按钮 W1 时，D1 点亮，电路如图 D-6 所示。

2. 程序结构合理、可读性强。

3. 将原文件转换为 OBJ 或 HEX 文件。

二、评分方法

1. 能正确编写软件程序得 35 分，写错一条指令扣 1 分；漏写一条指令扣 1 分。

2. 软件程序优化合理得 10 分，软件程序结构不合理扣酌情扣 1～10 分。

3. 能正确调试、产生 OBJ 或 HEX 文件得 5 分，产生的 OBJ 或 HEX 文件不正确酌情扣 2～4 分；没有产生 OBJ 或 HEX 文件扣 5 分。编写、调试软件程序评分表见下表所列。

编写、调试软件程序评分表

项　　目	总　　分	得　　分
编写软件程序	35	
优化软件程序	10	
调试、产生 OBJ 或 HEX 文件	5	

时间：_____　考评员：_____　评分：_____

第三部分　烧写软件（10 分）

一、考试要求

将 OBJ 或 HEX 文件正确烧写入芯片。

二、评分方法

能将产生的 OBJ 或 HEX 文件正确烧写入芯片得 10 分，没有烧写入扣 10 分；烧写入错误酌情扣 1～10 分。软件烧写评分表见下表所列。

软件烧写评分表

项　　目	总　　分	得　　分
烧写软件	10	

时间：_____　考评员：_____　评分：_____

单片机装调工专项能力认证试题中级技能试题 7

姓名：_____ 准考证号：_____ 成绩：_____

第一部分　元器件整形焊接（40 分）

一、考试要求

1. 将需要焊接的元器件整形。
2. 图 D-7 所示为彩灯电路，根据图 D-7 焊接电路。

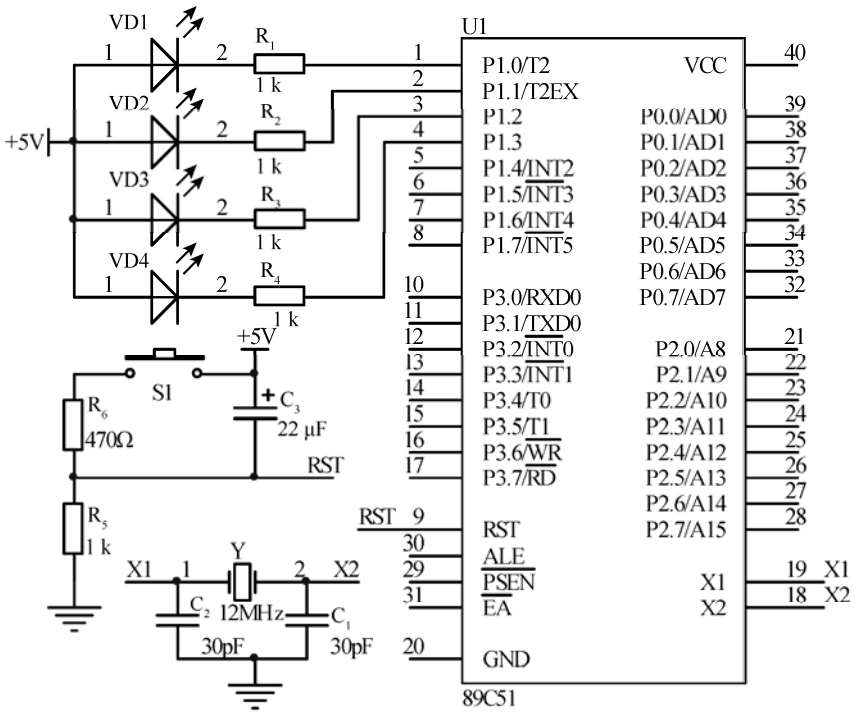


图 D-7　彩灯电路

二、评分方法

1. 元器件整形正确、美观得 5 分；整形不规范，一个元器件扣 0.5 分。

2. 元器件布局合理得 10 分；元件布局不合理扣 1～10 分。

3. 能正确焊接，且焊点光滑美观得 25 分，漏焊一个扣 1 分；虚焊一个扣 0.5 分；焊点一处不光滑或有毛刺扣 0.5 分。元器件整形焊接评分表见下表所列。

元器件整形焊接评分表

项　目	总　分	得　分
元器件整形	5	
元器件布局	10	
元器件焊接	25	

时间：_____　考评员：_____　评分：_____

第二部分　编写、调试软件程序（50 分）

一、考试要求

1. 编写软件程序，要求所有发光二极管先闪烁两次（亮灭时间均为 1 秒），然后从上亮到下（点亮时间为 1 秒），。

2. 程序结构合理、可读性强。

3. 将原文件转换为 OBJ 或 HEX 文件。

二、评分方法

1. 能正确编写软件程序得 35 分，写错一条指令扣 1 分；漏写一条指令扣 1 分。

2. 软件程序优化合理得 10 分，软件程序结构不合理扣酌情扣 1～10 分。

3. 能正确调试、产生 OBJ 或 HEX 文件得 5 分，产生的 OBJ 或 HEX 文件不正确酌情扣 2～4 分；没有产生 OBJ 或 HEX 文件扣 5 分。编写、调试软件程序评分表见下表所列。

编写、调试软件程序评分表

项　目	总　分	得　分
编写软件程序	35	
优化软件程序	10	
调试、产生 OBJ 或 HEX 文件	5	

时间：_____　考评员：_____　评分：_____

第三部分　烧写软件（10 分）

一、考试要求

将 OBJ 或 HEX 文件正确烧写入芯片。

二、评分方法

能将产生的 OBJ 或 HEX 文件正确烧写入芯片得 10 分，没有烧写入扣 10 分；烧写入错误酌情扣 1～10 分。软件烧写评分表见下表所列。

软件烧写评分表

项　　目	总　　分	得　　分
烧写软件	10	

时间：_____　考评员：_____　评分：_____

单片机控制与应用实训教程

结合教学需要，精心设计任务项目

注重工程实践，巧妙拓展技能训练

ISBN 978-7-121-10784-9

上架建议 电子技术

策划编辑：谭佩香
责任编辑：鄂卫华
封面设计：王 嵩

天启星
http://www.tqxbook.com

定价：28.00元