

## 目录

概述 .....	3
第一章TPC-ZK-USB实验系统介绍 .....	4
概述 .....	4
第二章 TPC-ZK-USB实验系统硬件环境 .....	5
2.1 USB模块介绍 .....	5
2.1.1 USB模块结构 .....	5
2.1.2 USB模块功能 .....	5
2.1.3 USB模块的对外接口 .....	5
2.1.4 USB模块跳线说明 .....	5
2.1.5 USB模块的安装 .....	6
2.2 TPC-ZK实验系统结构及主要电路 .....	11
2.2.3 用户扩展实验区 .....	16
2.2.4 实验台跳线开关 .....	16
2.2.5 20芯双排插座、26芯双排插座 .....	16
2.2.5 直流稳压电源 .....	16
2.2.6 TPC-ZK实验系统开关及跳线说明 .....	17
第三章 环境安装及使用说明 .....	18
3.1 HQFC集成开发环境安装 .....	18
3.2 HQFC集成开发环境的使用说明 .....	20
3.2 HQFC集成开发环境下VC程序的使用说明 .....	31
第四章 基本实验 .....	36
实验一 I/O地址译码 .....	36
实验二 简单并行接口 .....	38
实验三 可编程并行接口8255 .....	41
实验四 七段数码管 .....	43
实验五 键盘显示控制实验 .....	47
实验六 竞赛抢答器 .....	53
实验七 交通灯控制实验 .....	56
实验八 可编程定时器 / 计数器 (8254) .....	59
实验九 继电器控制 .....	63
实验十 存储器读写实验 .....	66
实验十一 DMA传送 .....	69
实验十二 扩展DMA控制器8237 .....	76
实验十三 中断 .....	80
实验十四 扩展中断控制器8259 .....	88
实验十五 可编程并行接口8255方式 1 .....	93
实验十六 串行通讯8251 .....	97

实验十七	数/模转换器 .....	101
实验十八	模/数转换器0809.....	104
实验十九	步进电机控制实验 .....	109
实验二十	直流电机转速控制实验 .....	113
实验二十一	双色点阵发光二极管显示实验 .....	117
实验二十二	128X64字符图形液晶显示实验 .....	127
实验二十三	集成电路测试 .....	131
实验二十四	电子琴 .....	134

# 概述

**微**机原理与接口技术和**单**片机与接口是高等院校理工科类各专业的一门重要的计算机技术基础课程。随着计算机软硬件的不断升级换代和微机技术的广泛应用，微型计算机教学内容也随之更新，这就对相应的教学实验设备提出了新的要求。为此我公司总结过去十几年设计生产微机接口和单片机与接口等实验装置的经验，综合各学校讲课及实验老师的意见之后推出《TPC-ZK教学实验系统》新产品。该仪器适应能力更强，配置更灵活。该实验系统可以配接不同的核心板，成为不同的实验接口系统。

## 一、TPC-ZK教学实验系统主要特点：

- ★ 根据学校不同的需求，可以配接PCI卡、USB接口、各类单片机等核心板。构成不同的接口实验系统。TPC-ZK实验系统可以同时配接微机接口（PCI微机接口或USB微机接口）和其它类型的接口核心板（51单片机等）二种核心板。二种核心板可以通过开关SW2选择手动选择。也可以自动优先级选择，即插上实验系统板上的核心时就自动断开实验系统板下的核心板。方便老师习惯选择核心板。
- ★ 实验台结构采用了综合实验和扩展实验模块相结合的方式，既保证基本实验结构紧凑，实验方便又有扩展实验灵活的特点。
- ★ 实验接线采用8芯排线和单根自锁紧导线相结合的方式，插线方便灵活。
- ★ 接口实验增加了实用性、趣味性的项目，使用汇编语言和C语言编写实验的程序。
- ★ 实验系统基本实验包括：8255并行接口实验模块；8254可编程定时器/计数器实验模块（书中部分图片说明标识为8254）；8251串行异步通信实验模块；8259中断控制器实验模块；AD0809模数转换实验模块；DA0832数模转换实验模块；RAM6264存储器实验模块；8237DAM控制器实验模块等。
- ★ 扩展实验模块包括：8279键盘显示控制器实验模块；LCD字符图形液晶显示模块；红外收发实验模块；无线收发实验模块；16X16LED点阵显示模块；红外、压力、温度、湿度传感器实验模块；16650串行异步通信、简单I/O扩展实验模块；FPGA实验模块等。（陆续增加中）
- ★ 核心控制板包括：51系列单片机模块；PCI微机接口模块；USB微机接口模块；80386微机接口模块；C8051单片机；PSOC现场可编程系统等。（陆续增加中）
- ★ 微机接口集成开发环境，支持WIN2000、WINXP等操作系统。可以方便的对程序进行编辑、编译、链接和调试，可以查看实验原理图，实验接线，实验程序进实验演示。可以增加和删除自定义实验项目。
- ★ 实验程序可以使用宏汇编和C语言，集成实验开发软件可以自动识别汇编语言还是C语言源程序，可以对汇编程序和C语言程序进行调试。
- ★ 实验系统PCI微机接口备有32位数据可扩展模块（可选），可以完成32位数据实验。
- ★ 实验台有二个扩展接口，非常方便用户进行扩展块实验和扩展实验开发与设计。扩展接口采用20芯和26芯排线连接，接插非常方便。

# 第一章TPC-ZK-USB实验系统介绍

## 概述

在各种计算机外围接口不断推陈出新的今天，USB接口已经成为个人计算机最重要的接口方式之一，USB接口设备的应用也以惊人的速度发展，几乎新型的PC都100%支持USB技术。了解和掌握USB的应用及开发是计算机类、电子类、物理类本科生、大专生的新课题。

TPC-ZK-USB微机接口实验系统正是在这种背景下推出的。该设备在TPC-ZK实验系统上配置了USB接口模块，直接与主机(PC)的USB接口连接，形成了一套完整的USB接口的微机接口实验系统。该系统适应当前高等院校所开设的《微机原理及其应用》和《微机接口技术》这两门课的实验，同时也提供了最新接口USB的实验，使学生在校学习期间不仅有机会接触常规接口，同时有机会接触新型的接口，为学生们今后从事微机开发应用打下基础。

### 1.2 TPC-ZK-USB实验系统构成及特点

该系统由一块USB总线接口模块、TPC-ZK实验系统及集成开发环境软件组成。USB总线接口模块通过USB总线电缆与PC机相连，模块直接插在TPC-ZK实验系统上。其主要特点如下：

1. USB总线接口使用ISP1581/1583 USB2.0高速接口芯片，完全符合USB2.0规范。提供了高速USB下的通信能力，即插即用。

2. 满足《微机原理与接口技术》课程教学实验要求。实验台接口集成电路包括：可编程定时器/计数器（8254）、可编程并行接口（8255）、数/模转换器(DAC0832)、模/数转换器(ADC0809)等。外围电路包括：逻辑电平开关、LED显示、七段数码管显示、8X8双色发光二极管点阵及驱动电路、直流电机步进电机及驱动电路、电机测速用光藕电路、继电器及驱动电路、喇叭及驱动电路、键盘显示控制电路等。

3. 在USB接口模块上扩展有DMA控制器8237，可以完成微机DMA传送以及USB的DMA传送等实验。

4. 开放式结构，模块化设计支持开放实验。实验台上除固定电路外还设有用户扩展实验区。插座引脚都有对应的“自锁紧”插孔，利用这些插孔可以搭试更多的自己设计的实验，方便的进行课程设计。

5. 功能强大的软件集成开发环境，支持Win2000;WinXP 等操作系统（不支持WIN98系统）。可以方便的对程序进行编辑、编译、链接和调试，可以查看实验原理图，实验接线，实验程序并进行实验演示。可以增加和删除实验项目。

6. 实验程序可以使8086汇编和C语言编程实验。可以对汇编程序和C语言程序进行调试(C语言调试系统需安装了VC软件,因版权不提供该软件)。

7. 系统还提供：字符、图形液晶显示实验模块；红外收发实验模块；无线通信实验模块；8279键盘显示实验模块等多种扩展实验模块。

8. 实验台自备电源，具有电源短路保护确保系统安全。

9. 使用USB接口与PC机相连，省却了打开主机箱安装接口卡的麻烦。

## 第二章 TPC-ZK-USB实验系统硬件环境

### 2.1 USB模块介绍

#### 2.1.1 USB模块结构

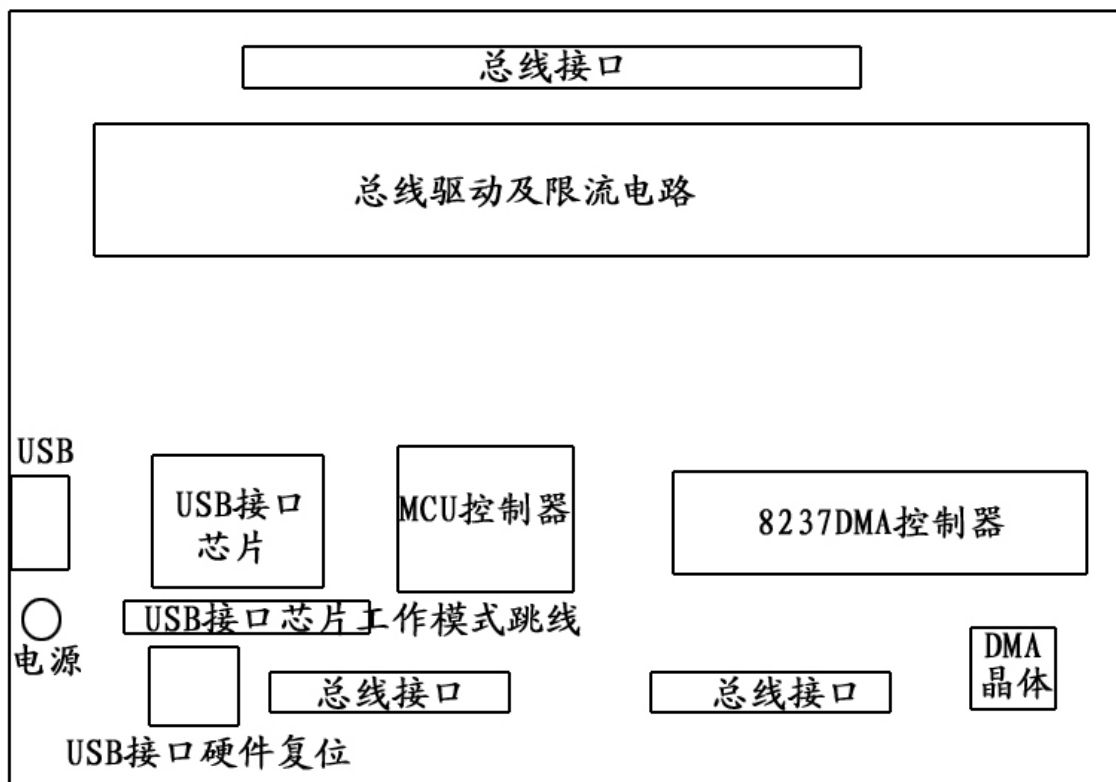


图2-1 USB模块结构图

#### 2.1.2 USB模块功能

1. 实验系统中的USB模块使用PHILIHPS的ISP1581/1583 USB2.0高速接口芯片，符合USB2.0接口规范，提供了高速USB下的通信能力。
2. 模块内扩展有DMA控制器8237，可以完成微机DMA传送和USB的DMA传送实验。
4. 该模块产生的仿ISA总线信号连到TPC-ZK实验系统上。

#### 2.1.3 USB模块的对外接口

1. 在该模块的右侧提供四个对外接口：
  - ①USB接口，连接到主机，实验时用于信息和数据的通信。
  - ②清零按钮（RESET），用于对USB接口模块内部电路的初始化。
2. 在模块的上下两侧提供三个对外接口：
  - ①50芯接口，为实验台提供仿ISA总线信号。信号安排与TPC-ZK实验系统上50芯信号插座信号一一对应。
  - ②两个20芯接口，连接到TPC-ZK实验系统上所需电源与信号。

#### 2.1.4 USB模块跳线说明

在USB模块内，用一部分跳线选择ISP1581/1583和其它芯片的工作模式，跳线的连接说明

如下：

跳线管脚：

1	2	3
---	---	---

（电路板正向面向自己）

JP1: MODE1 ISP1581/1583 ALE/A0 功能选择。

2—3短接 低电平 选择ALE功能（地址锁存使能）

1—2短接 高电平 选择A0功能（地址数据指示）

（USB模块出厂时选择2—3短接）

JP2: M0/DA1 选择ISP1581/1583在通用处理器模式下的读写功能。

2—3短接 低电平 选择Motorola 类型的微处理器

1—2短接 高电平 选择8051 类型的微处理器

（USB模块出厂时选择1—2短接）

JP3: BUS/DA0 选择ISP1581/1583 总线模式

2—3短接 低电平 选择断开总线模式，AD[7:0]多路复用

1—2短接 高电平 选择通用处理器模式，AD[7:0]8位地址线

（USB模块出厂时选择2—3短接）

JP4: ISP1581/1583 片选信号选择

1—2短接 ISP1581/1583 片选信号由MCU 产生

2—3短接 ISP1581/1583 片选信号由地址译码产生

（USB模块出厂时选择2—3短接）

JP7: DMA控制器时钟选择

2—3短接 选择振荡器产生时钟(4MHZ时钟)

1—2短接 选择由MCU 产生时钟

（USB模块出厂时选择2—3短接）

## 2.1.5 USB模块的安装

安装步骤如下：

1. 关上实验台电源。

2. 将USB模块插入TPC-ZK实验系统核心区接口上。（注意方向）。

3. USB电缆的一端接模块的USB口，另一端接主机USB口。

4. 打开实验台电源。

5. 系统将自行检测到模块的接入，选择用户光盘上的USB驱动程序完成驱动的安装。

安装驱动过程如下：

USB电缆接入主机，连接USB模块并加载电源后，系统将自行检测到模块的接入，第一次安装时，会提示用户发现新硬件并要求安装设备驱动：



图2-2 系统发现新硬件

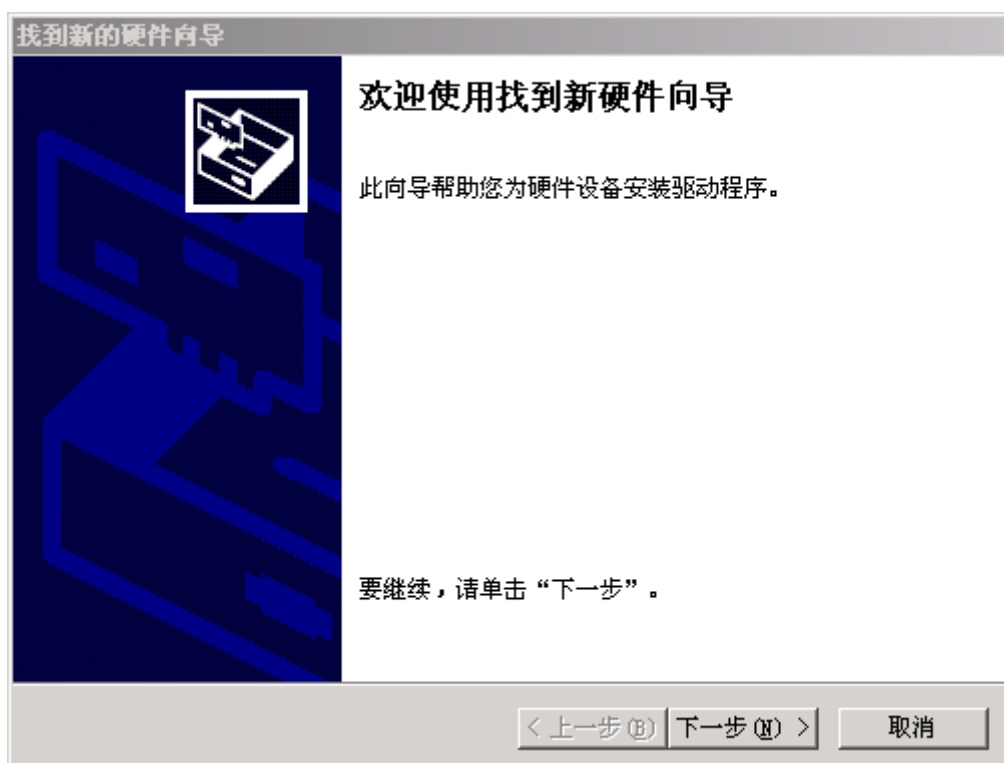


图2-3 提示找到新硬件

找到新硬件，需为此硬件指定设备驱动程序：

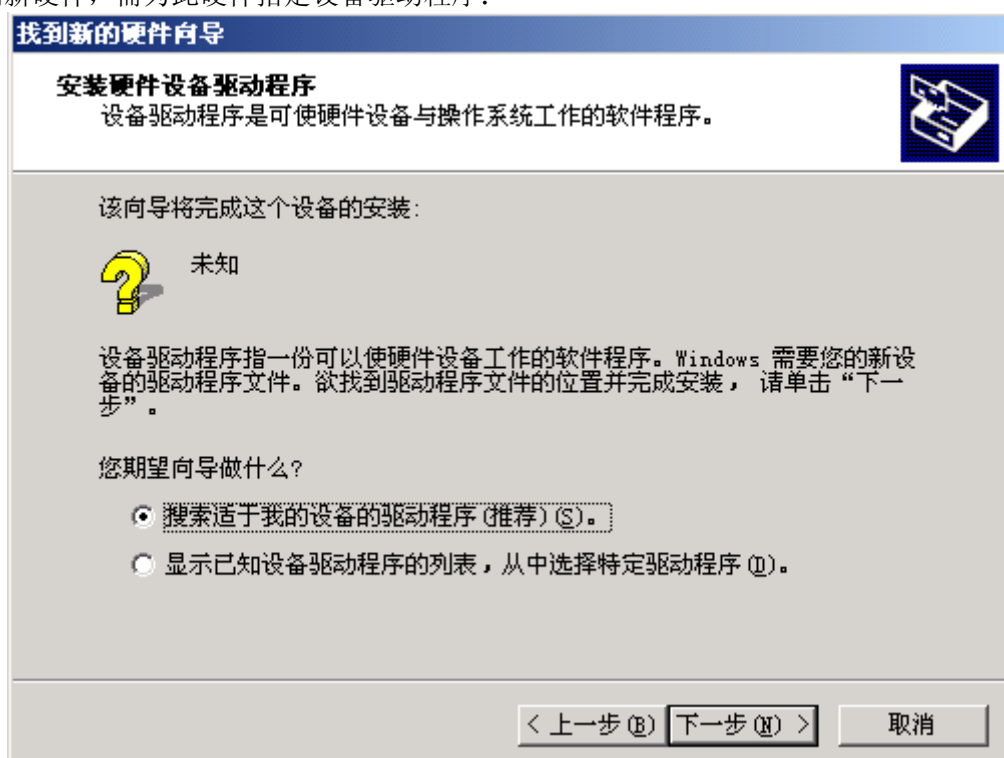


图2-4 提示按装驱动

选择驱动所在位置：(CD-ROM中driver目录下或指定驱动所在位置)

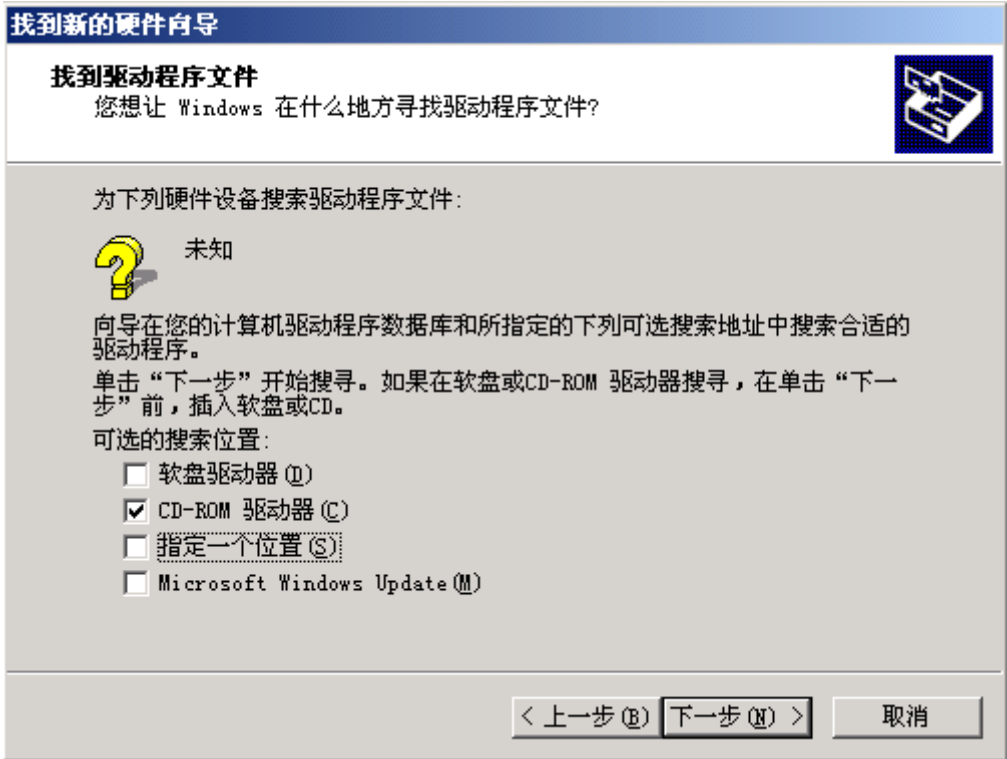


图2-5 指定驱动所在位置

浏览驱动所在位置并选定驱动安装信息文件TPCA. inf:

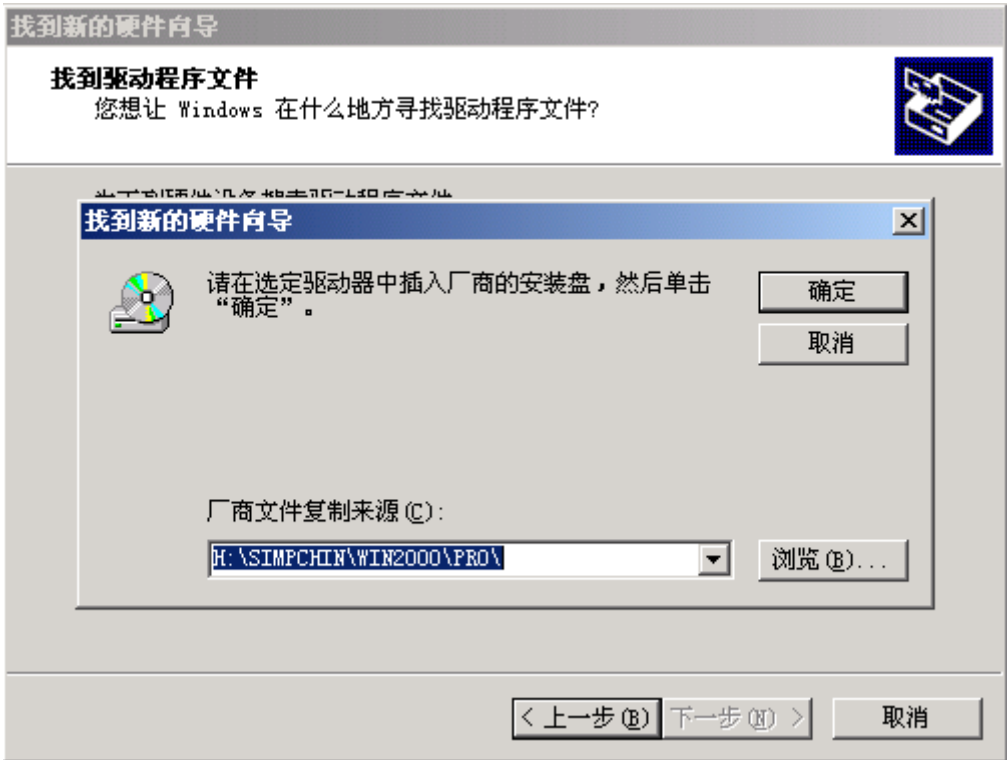




图2-6 浏览并找到驱动

选定TPCA.inf安装信息文件，并打开：

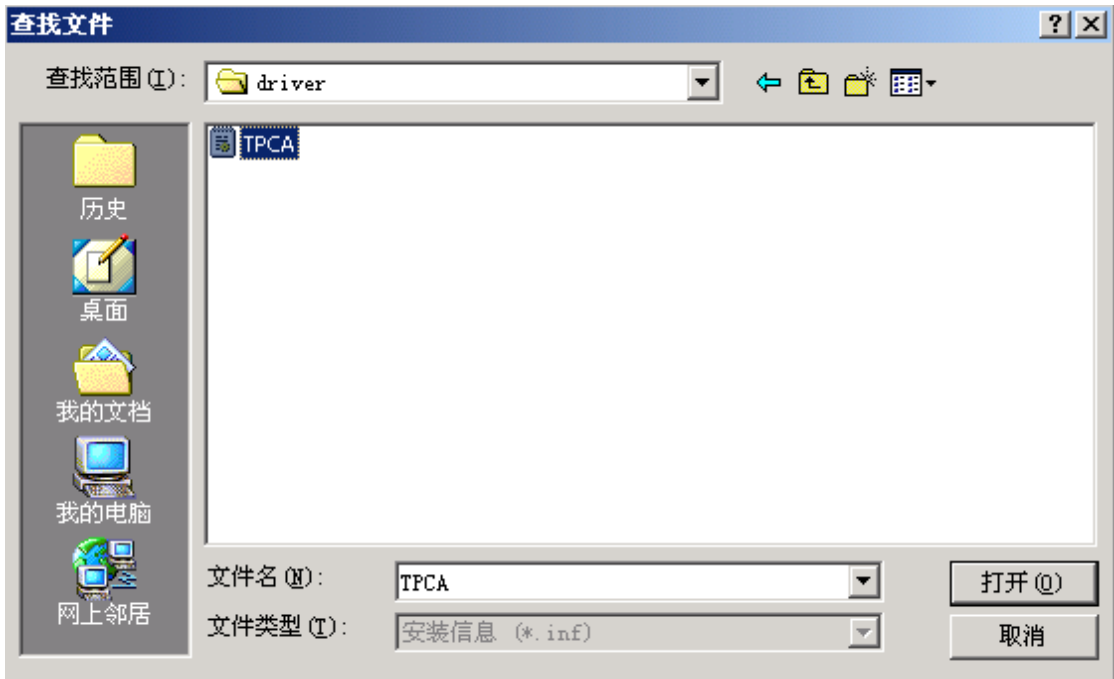


图2-7 找到驱动并选定

点击下一步，系统将自动为TPC设备安装其驱动：

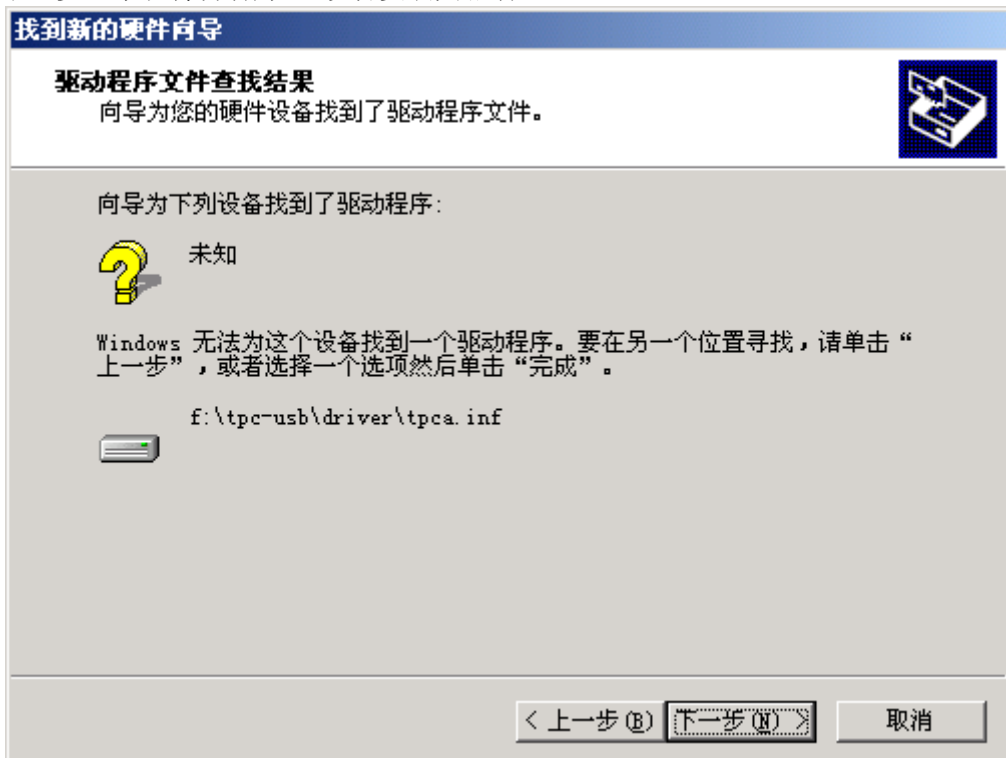


图2-8 安装驱动

驱动安装完毕：

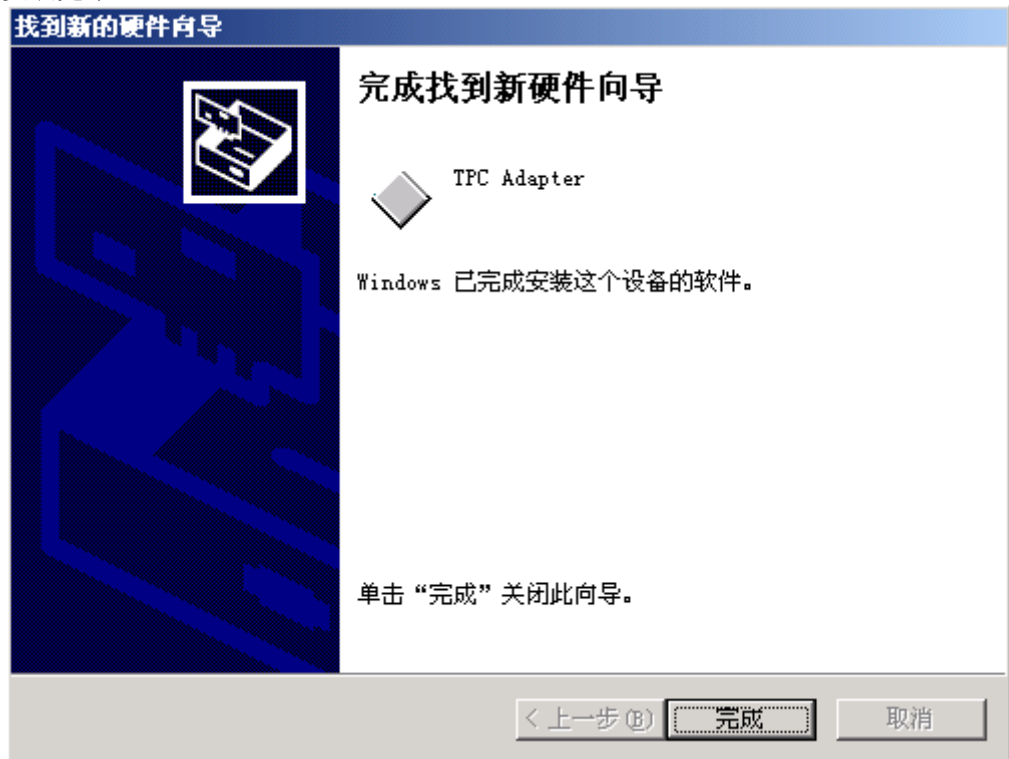


图2-9 完成安装

设备安装检测：

右键单击“我的电脑”，选择“属性”，选择硬件选项中的“设备管理器”，即可在通用串行总线控制器中找到已安装的TPC Adapter设备。至此安装完毕。

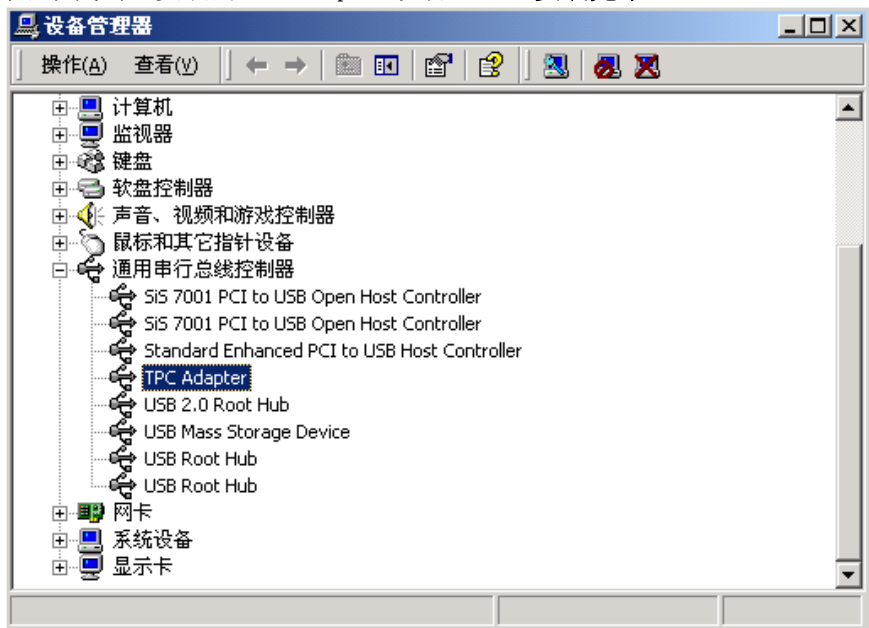


图2-10 查看安装

## 2.2 TPC-ZK实验系统结构及主要电路

### 2.2.1 TPC-ZK实验系统结构图

如图2-14

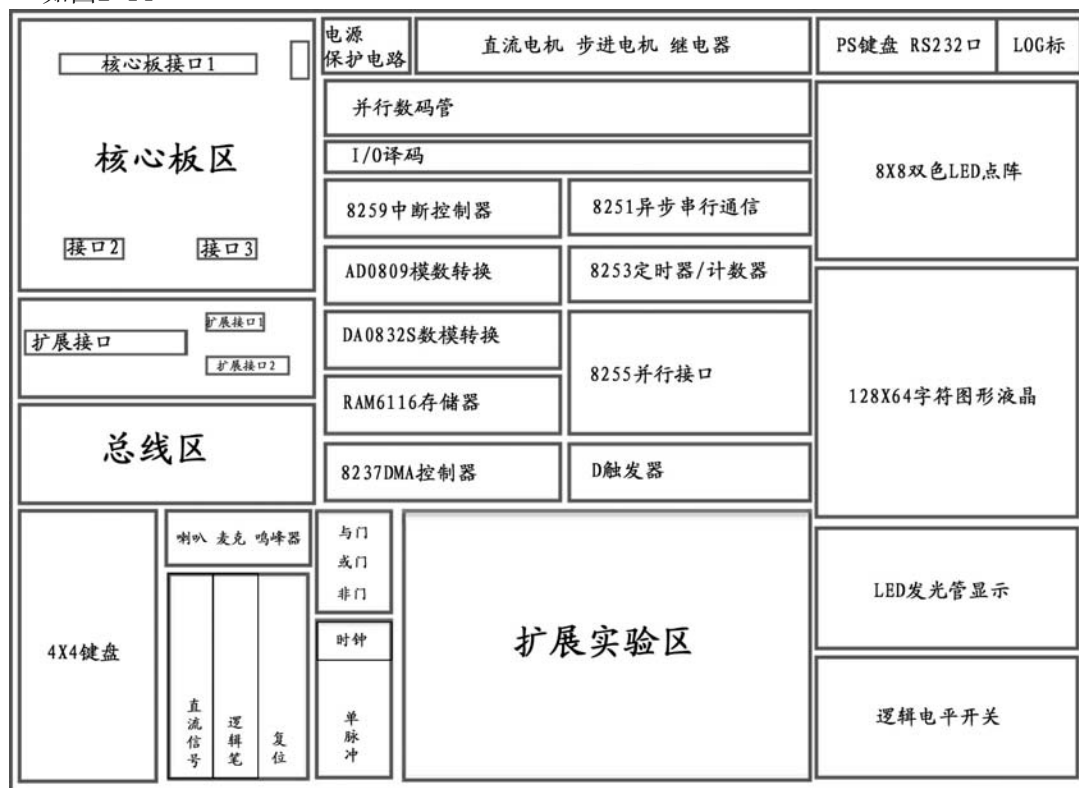


图2-14

### 2.2.2 实验台上包括的主要电路：

#### 1、50芯总线信号插座及总线信号插孔

1	+5V	11	E245	21	A7	31	A1	41	ALE
2	D7	12	IOR	22	A6	32	GND	42	T/C
3	D6	13	IOW	23	A5	33	A0	43	A16
4	D5	14	AEN	24	+12V	34	GND	44	A17
5	D4	15	DACK	25	A4	35	MEMW	45	A15
6	D3	16	DRQ1	26	GND	36	MEMR	46	A14
7	D2	17	IRQ	27	A3	37	CLK	47	A13
8	D1	18	+5V	28	-12V	38	RST	48	A12
9	D0	19	A9	29	A2	39	A19	49	A10
10	+5V	20	A8	30	GND	40	A18	50	A11

50芯总线信号插座在实验台左上方，总线插座信号安排如上表。各总线信号采用“自锁紧”插孔和8芯针方式在标有“总线”的区域引出，有数据线D0-D7、地址线A19-A0、I/O读写

信号IOR IOW、存储器读写信号 MEMR MEMW、中断请求 IRQ、DMA申请DRQ、DMA回答DACK、AEN等。

## 2、微机接口I/O地址译码电路

实验台上I/O地址选用280H—2BFH 64个，分8组输出：Y0—Y7，其地址分别为 280H—287H；288H—28FH；290H—297H；298H—29FH；2A0H—2A7H；2A8H—2AFH；2B0H—2B7H；2B8H—2BFH，8根输出线在实验台“I/O地址”处分别由自锁紧插孔引出。见图2-15

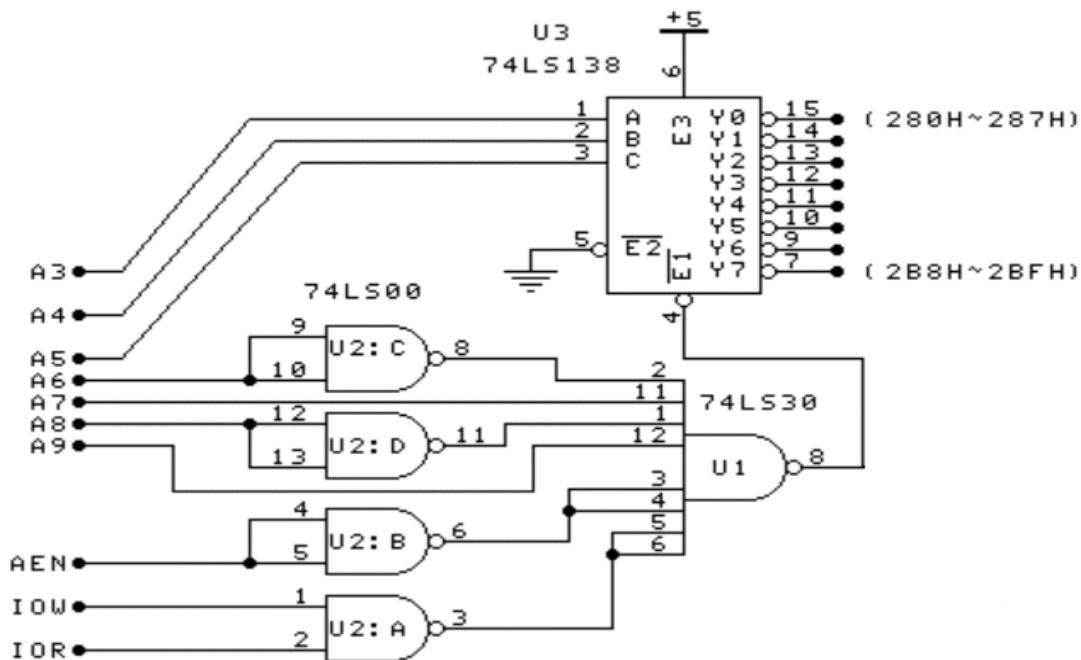


图2-15 I/O地址译码电路

## 3、时钟电路

如图2-16所示，输出1MHZ、2MHZ两种信号，供定时器/计数器、A/D转换器、串行接口实验使用。

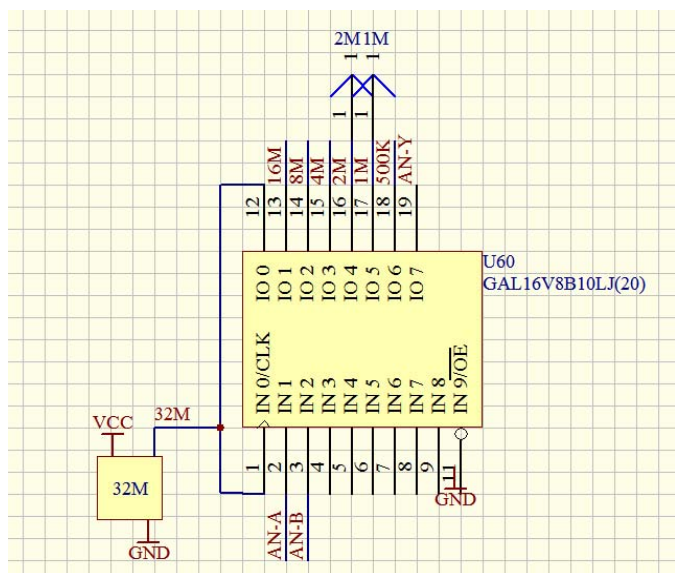


图2-16 时钟电路

#### 4、逻辑电平开关电路

如图2-17所示，实验台右方有8个开关K0-K7，开关拨到“1”位置时开关断开，输出高电平。拨到“0”位置时开关接通输出低电平。电路中串接了保护电阻，接口电路不直接同+5V、GND相连，有效的防止因误操作损坏集成电路现象。

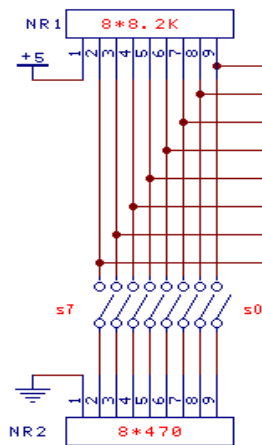


图2-17 逻辑电平开关电路

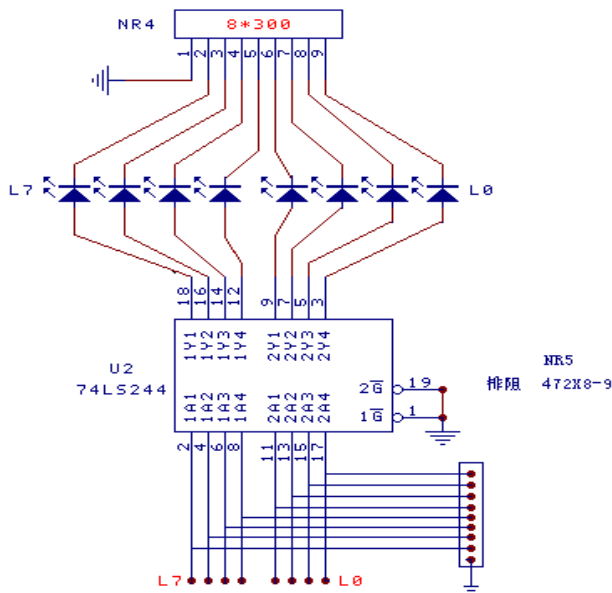


图2-18 发光二极管及驱动电路

#### 5、LED显示电路

如图2-18所示，实验台上设有8个发光二极管及相关驱动电路(输入端L7~L0)，当输入信号为“1”时发光，为“0”时灭。

#### 6、七段数码管显示电路

实验台设有4个共阴极数码管及驱动电路，电路图如图2-19（图中省去了S2、S3二位

数码管)。段码输入端：a、b、c、d、e、f、g、dp，位码输入端：S0、S1、S2、S3。

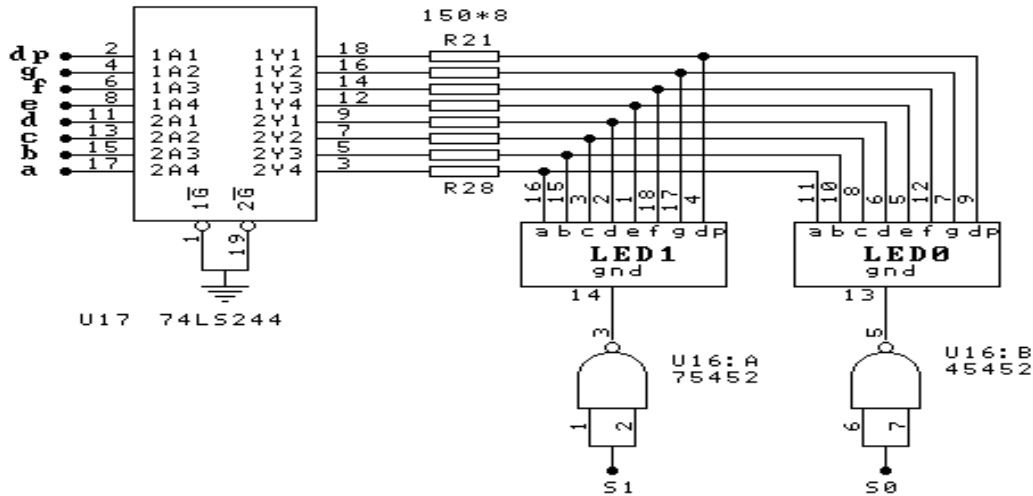


图2-19 数码管显示电路

### 7、单脉冲电路

如图2-20所示，采用RS触发器产生，实验者每按一次开关即可以从两个插座上分别输出一个正脉冲及负脉冲，供“中断”、“DMA”、“定时器/计数器”等实验使用。

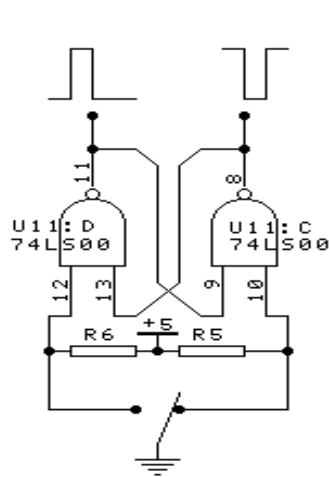


图2-20 单脉冲电路图

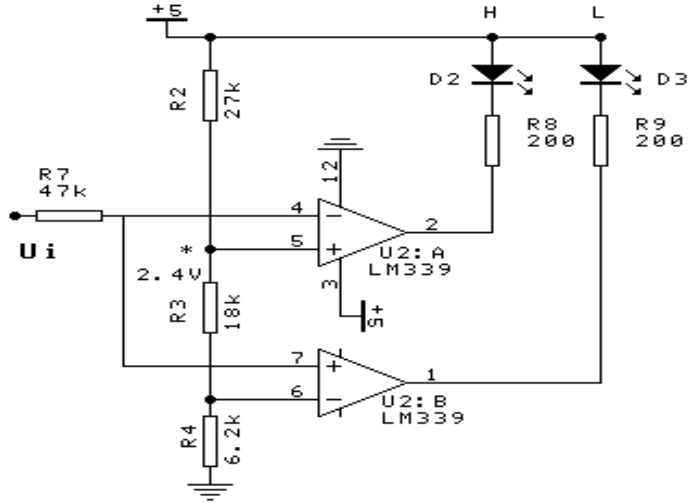


图2-21 逻辑笔电路

### 8、逻辑笔

如图2-21所示，当输入端Ui接高电平时红灯(H)亮，接低电平时绿灯(L)亮。有一脉冲时，黄灯亮一次，计数指示灯加1。可以测试TTL电平和CMOS电平。

### 9、继电器及驱动电路

图2-22为直流继电器及相应驱动电路，当其开关量输入端“Ik”输入数字量“1”时，继电器动作，常开触点闭合红色发光二极管点亮。输入“0”时继电器常开触点断开发光二极管灭。

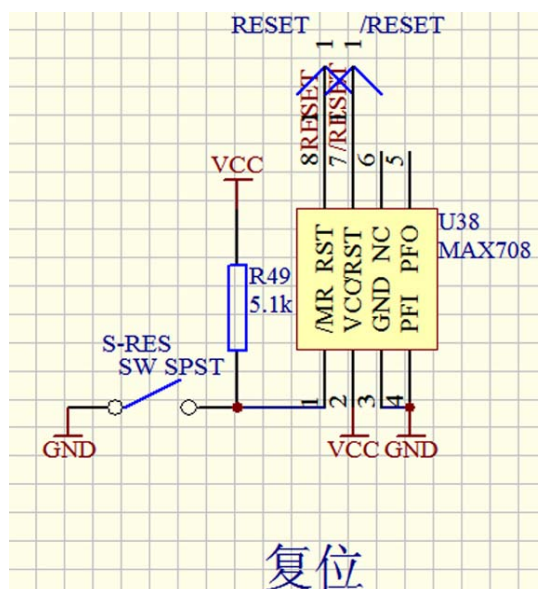
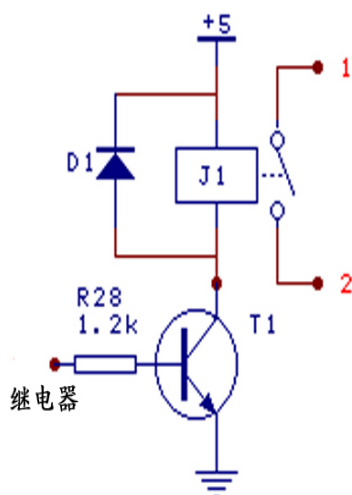


图2-22 继电器及驱动电路图

图2-23 复位电路

## 10、复位电路

图2-23为复位电路,实验台上有一复位电路,能在上电时,或按下复位开关RESET后,产生一个高电平和低电平两路信号供实验使用。

## 11、步进电机驱动电路

图2-24为步进电机的驱动电路，实验台上使用的步进电机驱动方式为二相励磁方式，BA、BB、BC、BD分别为四个线圈的驱动输入端，输入高电平时，相应线圈通电。

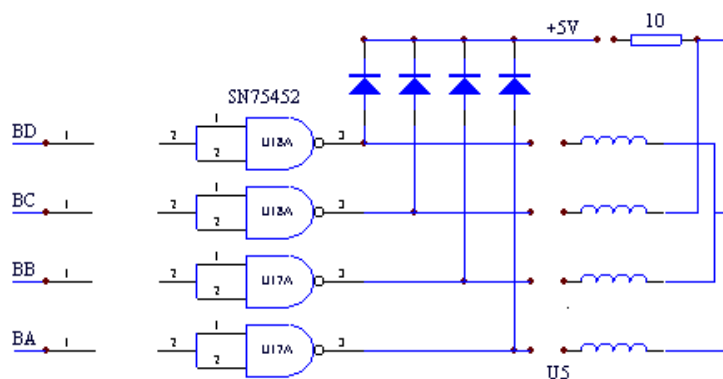


图2-24 步进电机驱动电路

## 12、接口集成电路

实验台上有微机原理及接口实验最常用接口电路芯片，包括：可编程定时器/计数器（8254）、可编程并行接口（8255）、数/模转换器（DAC0832）、模/数转换器（ADC0809）

串行异步通信（8251）、RAM存储器（6264）、中断控制器（8259）等，模块芯片与CPU相连的引线除去片选（CS）信号和每个实验模块特有信号外都已连好，与外围电路连接的关键引脚在芯片周围用“自锁紧”插座和8芯排线插针引出，供实验使用。

### 13、逻辑门电路

实验台上设有几个逻辑门电路。包括“与门”、“或门”、“非门”、“触发器”供实验时选择使用。

### 2.2.3 用户扩展实验区

实验台上设有通用数字集成电路插座，40芯活动插座以方便插拔器件。插座的每个引脚都用自锁紧插孔引出。实验指导书中所列出的部分实验（简单并行接口、集成电路测试等。这些电路也可选购为扩展实验模块）电路就是利用活动插座搭试的。扩展接口包括一个20芯的双排插座和一个26芯的双排插座，大板上基本信号都由该两个扩展接口插座引出，利用扩展接口可以进行其它的扩展模块实验。利用扩展插座及扩展接口可以进行数字电路实验，也可以设计开发新的接口实验或让学生做课程设计、毕业设计等项目。两个扩展接口信号安排见2.2.5介绍。

### 2.2.4 实验台跳线开关

为了方便实验，实验台上设有跳线开关，分以下几种：

3. +5V或+12V电源插针：为减轻+5V电源负载和各主要芯片的安全，及学生在学习中设置故障。在各主要实验电路附近都有相应的电源连接插针，当实验需要该部分电路时，用短路子短接插针即可接通电源。对用不到的电路可将短路片拔掉确保芯片安全。

### 2.2.5 20芯双排插座、26芯双排插座

实验台上有一个20芯双排插座JX1，用于外接附加的键盘显示实验板和其它用户开发的实验板。JX1各引脚信号安排如下：

2	4	6	8	10	12	14	16	18	20
GND	GND	1MHz	A1	A0	IOW	IOR	+5V	+5V	RESET
1	3	5	7	9	11	13	15	17	19
CS=2B0H	IRQ	D7	D6	D5	D4	D3	D2	D1	D0

26芯双排插座各引脚如下

2	4	6	8	10	12	14	16	18	20	22	24	26
-12V	GND	MEMW	DACK1	A3	A5	A7	A9	A11	8M	1M	CS=2B8H	+12V
1	3	5	7	9	11	13	15	17	19	21	23	25
+12V	+5V	MEMR	DRQ1	A2	A4	A6	A8	A10	32M	2M	/RESET	-12V

### 2.2.5 直流稳压电源

实验箱自备电源，安装在实验大板的下面，交流电源插座固定在实验箱的后测板上，交流电源开关在实验箱的右侧，交流电源开关自带指示灯，当开关打开时指示灯亮。在实验板右上角有一个直流电源开关，交流电源打开后再把直流开关拨到“开”的位置，直流+5V +12V -12V就加到实验电路上。

主要技术指标： 输入电压 AC 175—265V

输出电压/电流 +5V/2.5A +12V/0.5A -12V/0.5A



输出功率 25W

### 2.2.6 TPC-ZK实验系统开关及跳线说明

JCS1、JCS2：同时连接12时，选择其核心板方式为手动选择，即拨动核心控制板开关SW2选择是TPC-ZK实验系统大板上面的核心控制板还是大板下面的核心控制板（实验箱内）。同时连接23时，选择其核心板方式为自动优先极判断，即只要TPC-ZK实验系统大板上面核心区插入了核心控制板，就选择该核心板，自动断开大板下面（实验箱内）的核心板。

JCS3：选择逻辑笔测试输入信号是CMOS电平还是TTL电平。

JCS4：8X8LED点阵工作模式：

12短接时，工作于“非总线”模式。行信号、红色列信号、绿色列信号经过排线分别独立连接到LED点阵的行、红色列、绿色列上。

23短接时，工作于“总线”模式。行信号、红色列信号、绿色列信号经过LED总线D7~D0和选择信号分别写入行寄存器、红色列寄存器、绿色列寄存器上。

SW1：TPC-ZK实验系统直流电源开关，向上打开开关，向下关闭实验系统电源。

SW2：大板上核心板工作方式手动选择时，选择是实验系统大板上面的核心板，还是大板下面（实验箱内）的核心板。

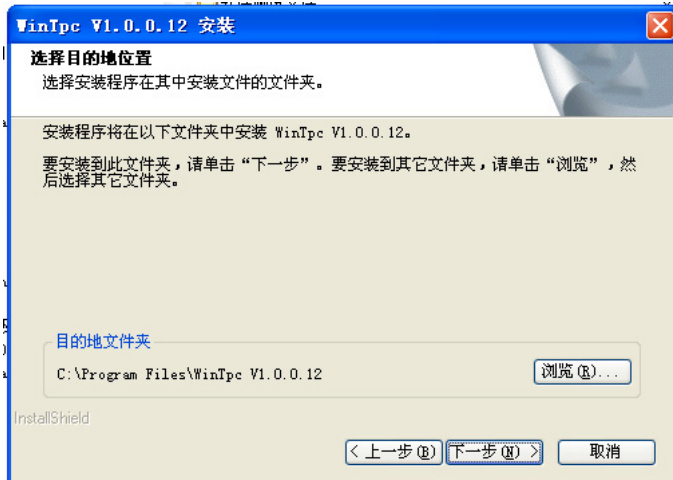
SW3：128X64字符图形液晶工作模式是并行模式还是串行模式。详细见128\*64字符图形液晶资料说明。

## 第三章 环境安装及使用说明

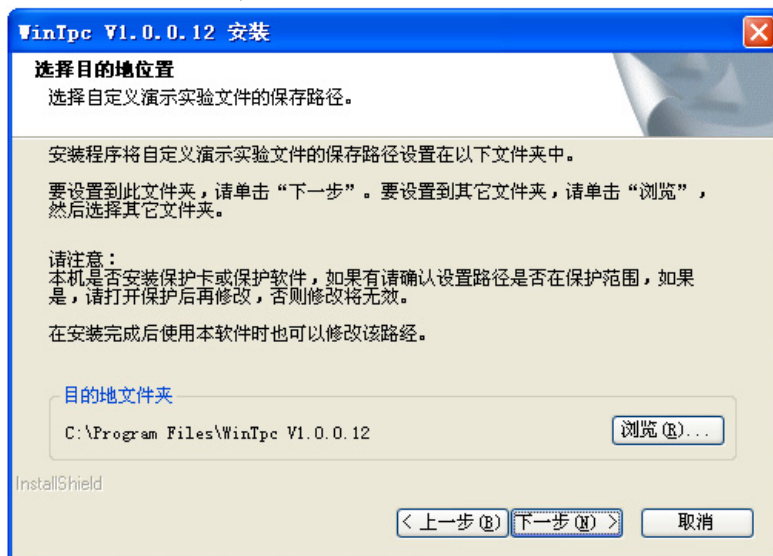
(HQFC集成开发环境适用于TPC系列教学实验系统)

### 3.1 HQFC集成开发环境安装

1)、点击光盘\HQFC集成开发环境\HQFC集成开发环境.EXE。如下图

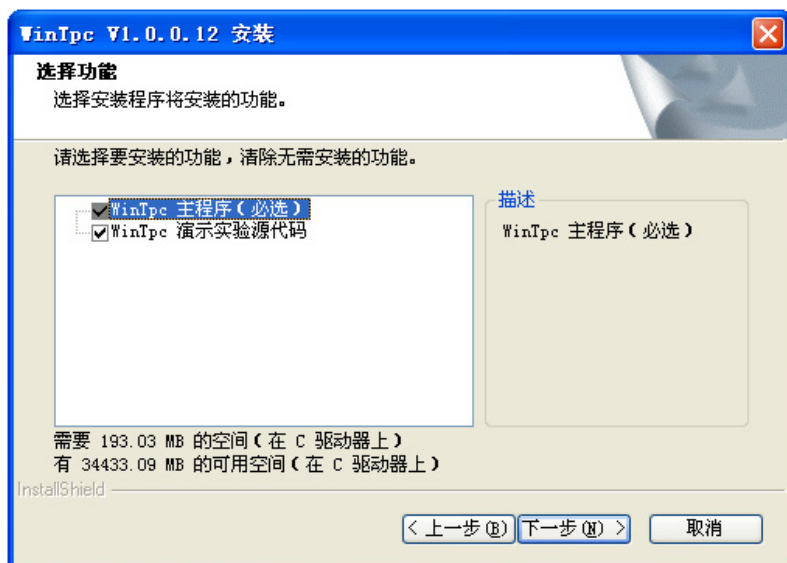


2)、选择软件安装路径后, 点击“下一步”。如下图

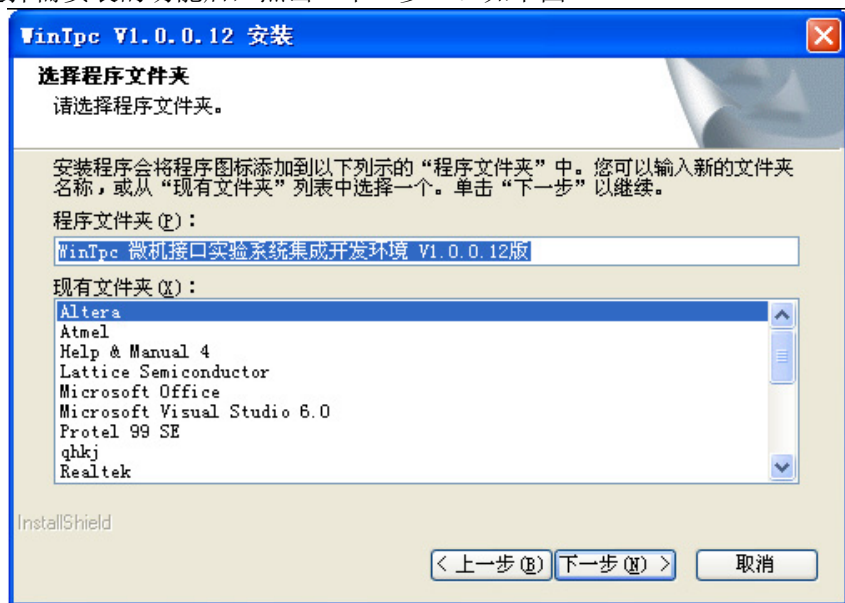


3)、选择自定实验放置路径, 如果自定义实验项目, 需要修改或调整, 请确认PC机操作系统是否安装保护功能, 如果安装请将自定义实验路径设置在未保护区, 如果需要保护, 在每次修改或添加自定义实验项目时, 请打开操作系统的保护后再操作。

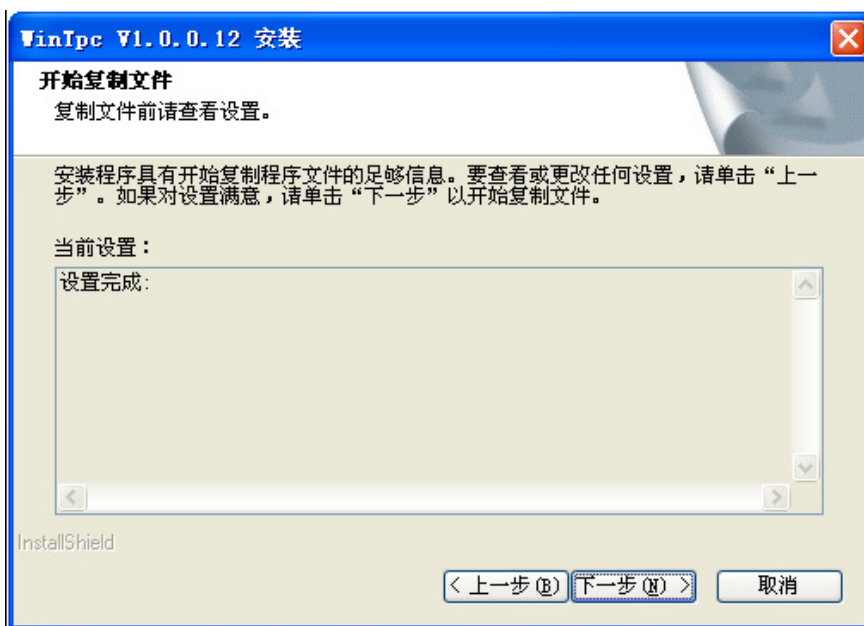
点击“下一步”。如下图



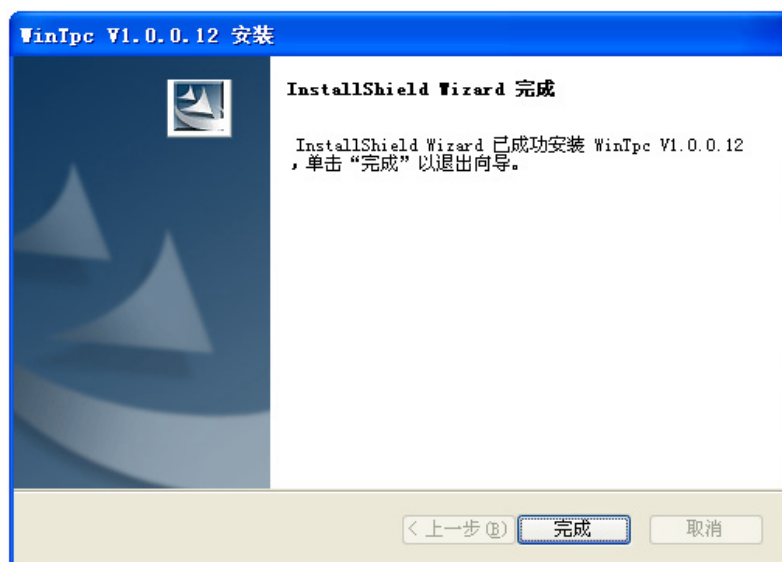
4)、选择需安装的功能后，点击“下一步”。如下图



5)、点击“下一步”。如下图



6)、点击“下一步”。如下图



7)、点击“安装”，便可安装该软件程序。如下图

8)、点击“完成”，该软件全部安装完。安装完后会在程序组中生成“HQFC集成开发环境”。

### 3.2 HQFC集成开发环境的使用说明

1、运行程序/“HQFC集成开发环境.EXE”，如下图

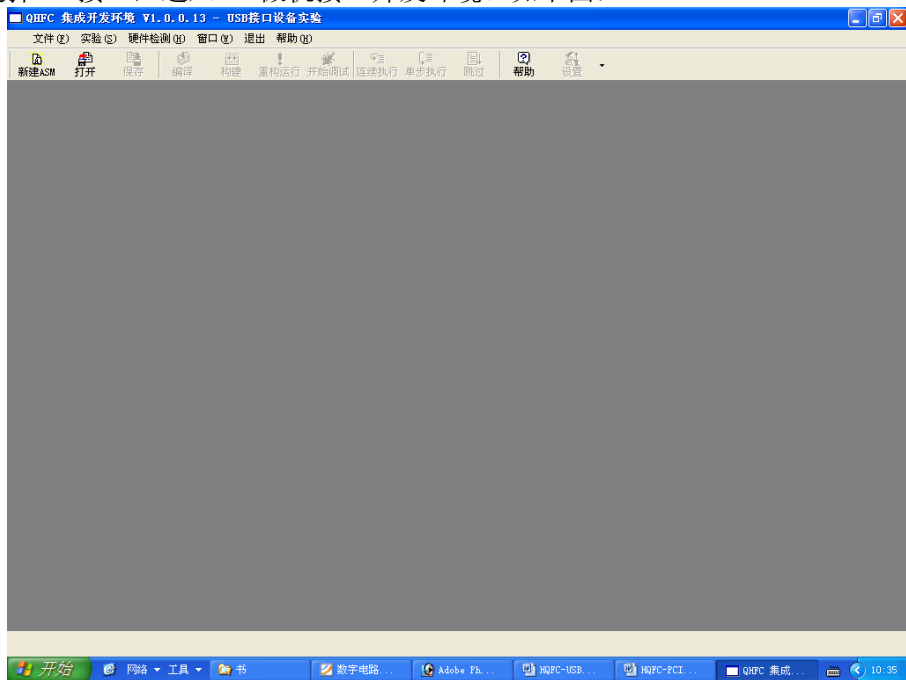


## 2、自动检测接口

软件自动检测所安装有的接口(包括PCI微机接口、USB微机接口、EX386嵌入微机接口)，如果检测到硬件显示为绿色，否则为红色。

## 3、选择接口类型

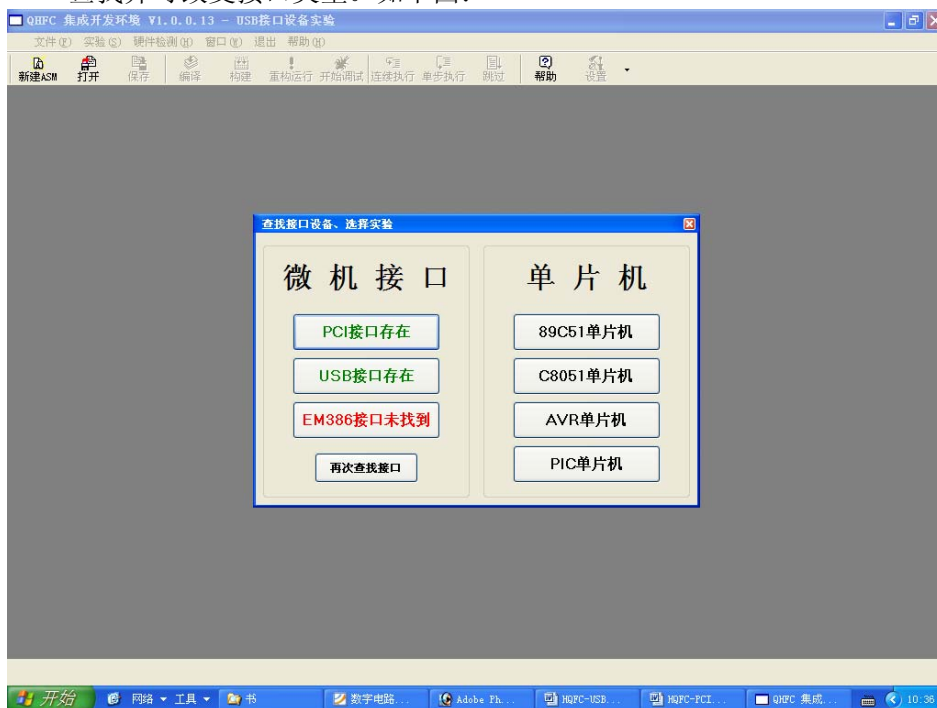
选择USB接口，进入USB微机接口开发环境。如下图：



## 4、硬件检测

查找并选择接口设备

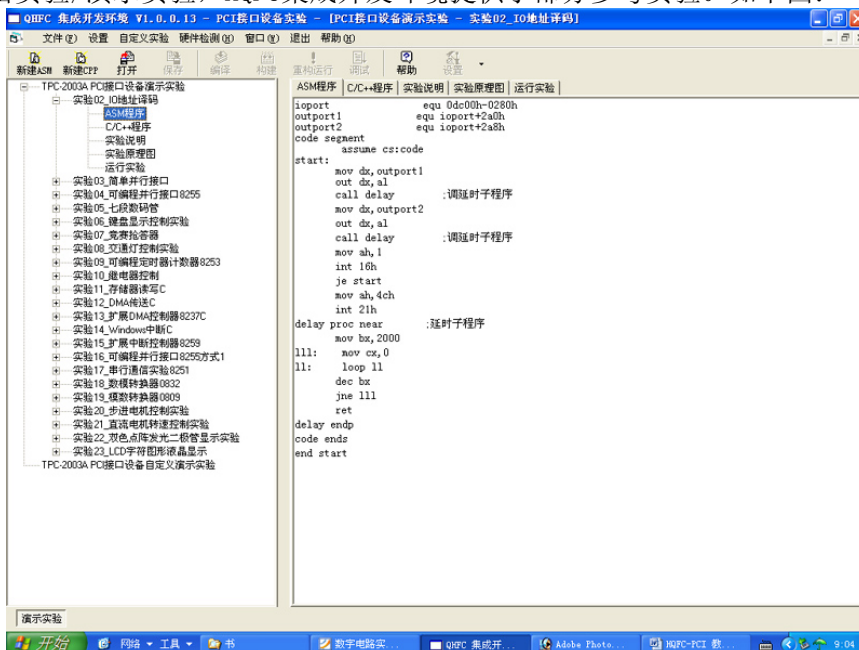
查找并可改变接口类型。如下图：



## 5、实验

### 1)、HQFC演示实验

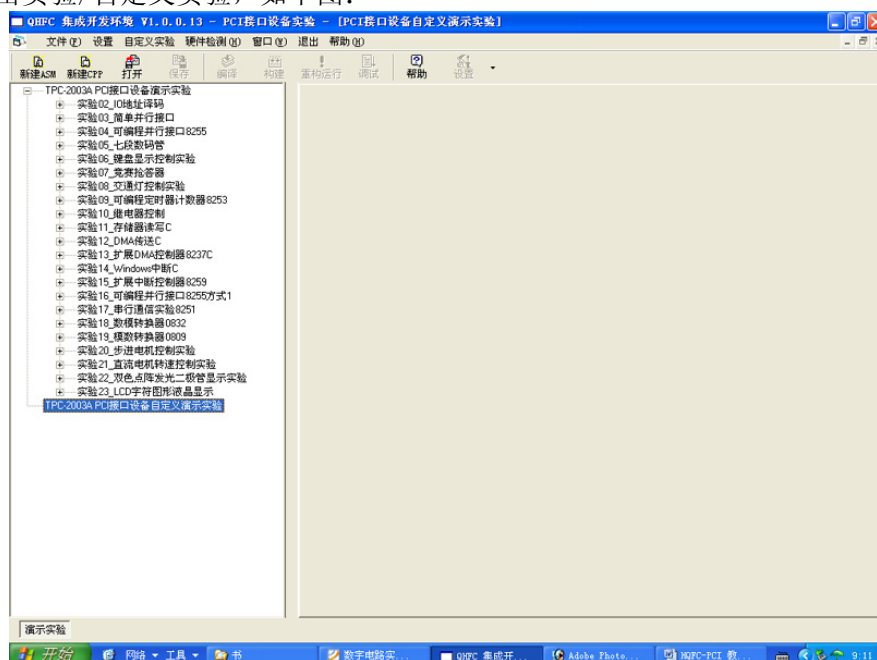
点击实验/演示实验，HQFC集成开发环境提供了部分参考实验。如下图：



演示实验包括实验的说明、原理图、部分源程序、部分运行程序等。点击窗口左边实验项目的子项，窗口右边显示相应内容，非常方便用户进行演示实验。

## 2)、自定义实验

点击实验/自定义实验，如下图：



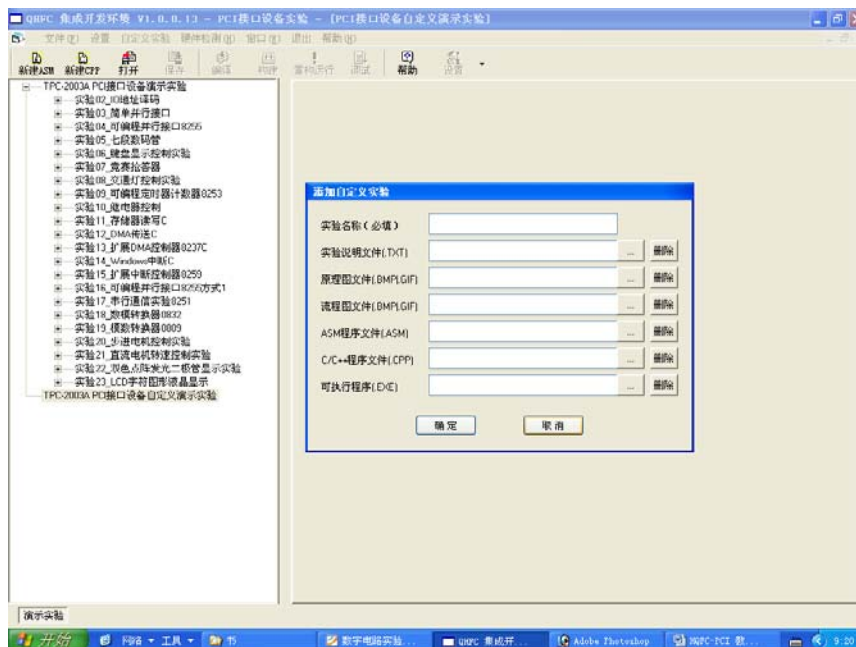
### a、设置

为了方便使用管理和使用用户自定义实验项目，可以设备自定义实验项目保存的路径。如果用户PC机安装了保护系统，为了时时修改和调整，可以该保存路径设在保护系统未保护的区域。为了不让其它用户修改和调整用户的自定义的实验项目，可以将保存路径设在保护系统的保护区域。

**说明：** 自定义实验路径为HQFC集成开发环境下所有接口类型的共用路径，修改过该路径后，软件将不能显示以前所设有自定义实验项目。

### b、添加自定义实验项目

点击“自定义实验”下的“添加自定义实验”，弹出添加窗口。如下图：



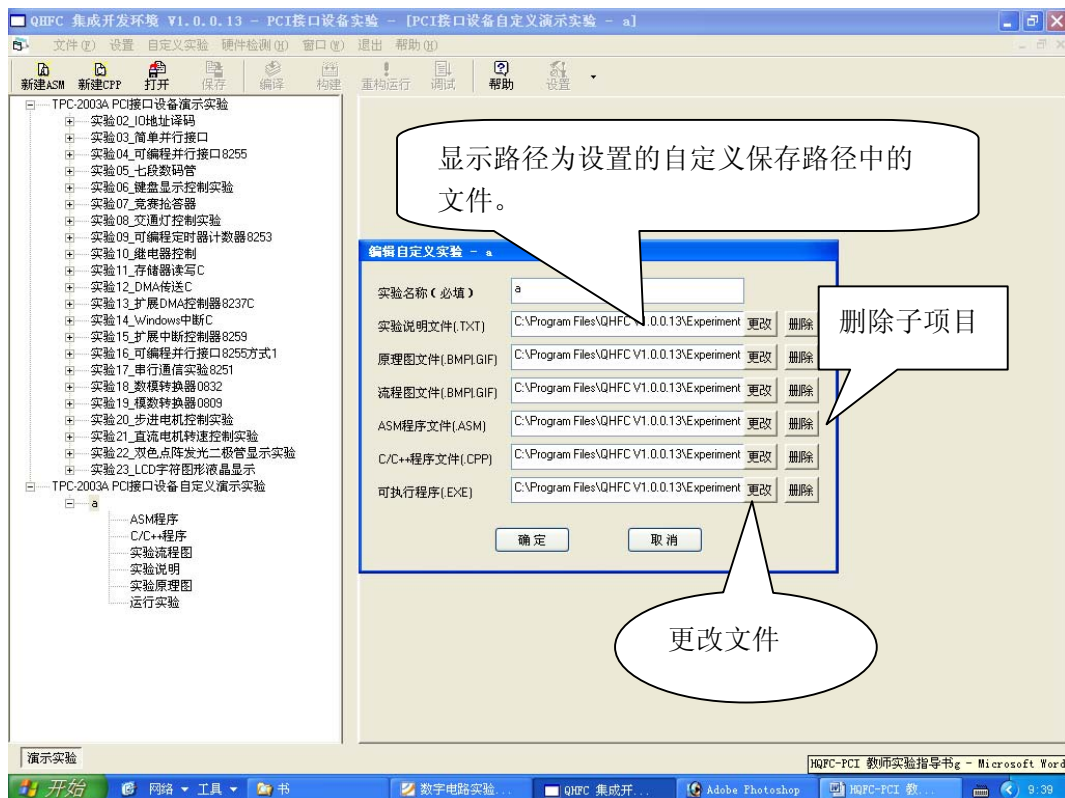
点击右侧的浏览选择编辑好的相应的文件，点击“确定”添加自定义实验项目，添加完成，在左侧HQFC演示实验下方将会出添加好的自定义项目。方便用户教学使用。

### c、编辑自定义实验项目

在窗口左边选中需要编辑的自定义实验项目，点击菜单中的自定义\编辑自定义实验或者单击鼠标右键选择弹出的菜单中的编辑自定义实验项目。修改和调整实验项目中的文件。

**提示：**直接修改自定义实验保存路径中的自定义项目中的文件内容时，请不要改变文件名，软件对文件名有一定的规则，否则软件将不能显示自定义项目中的子项目。





#### d、删除自定义实验项目

在窗口左边选中需要删除的自定义实验项目，点击菜单中的自定义\删除自定义实验或者单击鼠标右键选择弹出的菜单中的删除自定义实验项目。弹出提示是否删除窗口，如果需删除，选择是。

## 6、用户程序的编辑和编译

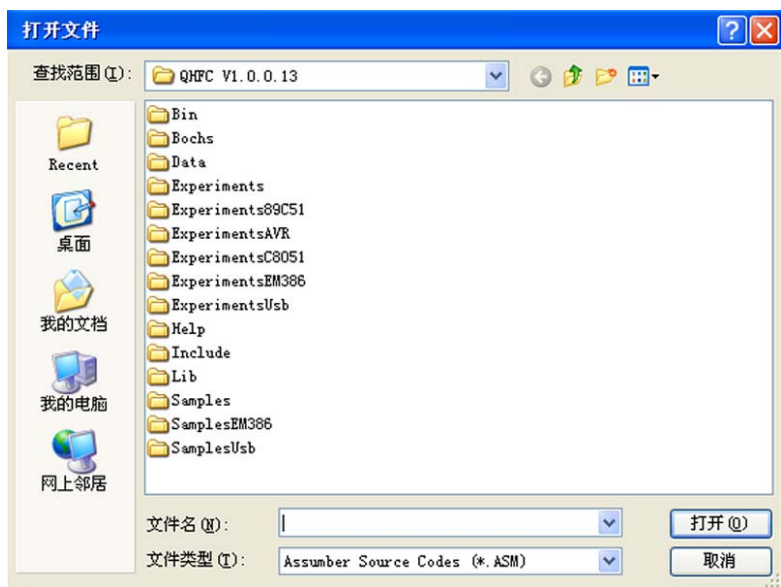
HQFC集成开发环境软件支持汇编程序(.asm文件)类型的程序开发。除了一般的编辑功能外，还有语法错误提示等功能。用户编辑好程序并保存后，即可方便地进行编译。

### 1. 新建一个源程序

在当前运行环境下，选择菜单栏中的“文件”菜单，菜单下拉后选择“新建”，或是在工具栏中单击“新建”快捷按钮，会出现源程序编辑窗口，建议用“另存为”为文件取名保存后，就新建一个“.asm”文件。

### 2. 打开一个源程序

当前运行环境下，选择菜单栏中的“文件”菜单，菜单下拉后选择“打开”，或是在工具栏中单击“打开”，会弹出“打开”文件选择窗口，“打开”窗口如图所示：



### 打开一个源程序

在窗口中“文件类型”下拉菜单中选择“ASM文档 (\*.asm)”一项，程序即显示当前目录下所有的asm文档，单击要选择的文件，选中的文件名会显示在“文件名”中，单击“打开”则打开当前选中的文档显示在文档显示区域。点击“取消”则取消新建源文件操作。

### 3. 编辑源程序

本软件提供了基本的编辑功能，并实现了实时的语法高亮，各项操作说明如下：

#### 撤销

当前运行环境下，选择菜单栏中的“编辑”菜单，菜单下拉后选择“撤销”，或是在工具栏中单击“撤销”，即可撤销上一步剪切或粘贴操作。

#### 剪切

当前运行环境下，选择菜单栏中的“编辑”菜单，菜单下拉后选择“剪切”，或是在工具栏中单击“剪切”，即可将文档显示区域中选中的内容剪切到剪贴板。

#### 复制

当前运行环境下，选择菜单栏中的“编辑”菜单，菜单下拉后选择“复制”，或是在工具栏中单击“复制”，即可将文档显示区域中选中的内容复制到剪贴板。

#### 粘贴

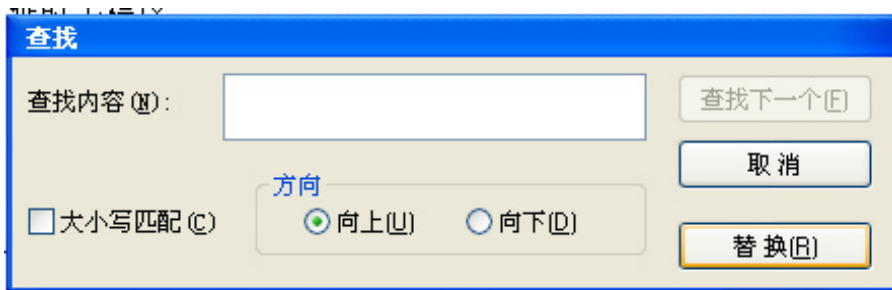
当前运行环境下，选择菜单栏中的“编辑”菜单，菜单下拉后选择“粘贴”，或是在工具栏中单击“粘贴”，即可将剪贴板中当前内容粘贴到文档显示区域光标所在处。

#### 全选

当前运行环境下，选择菜单栏中的“编辑”菜单，菜单下拉后选择“全选”，即可将文档区域中所有内容选中。

#### 查找

当前运行环境下，选择菜单栏中的“编辑”菜单，菜单下拉后选择“查找”，弹出查找对话框如图所示：

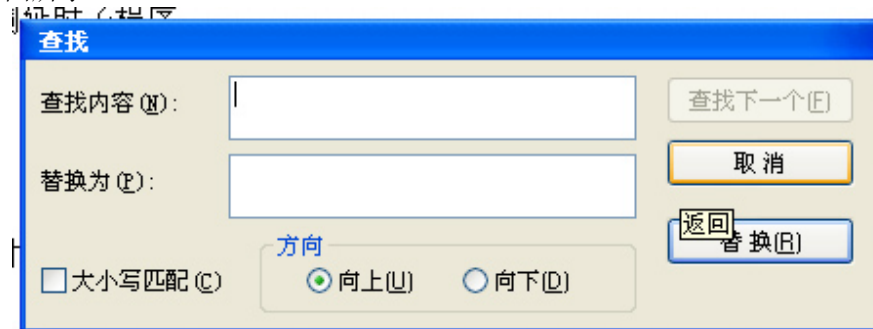


查找

在查找内容一栏中输入需要查找的内容，可选择“区分大小写”的查找方式，单击“查找下一个”程序则在文档显示区域中搜索与查找内容匹配的字符串，找到第一个后则高亮显示，用户点击查找下一个则继续搜索下一个匹配字符串，点击“取消”退出查找操作。

### 替换

当前运行环境下，选择菜单栏中的“编辑”菜单，菜单下拉后选择“替换”，弹出替换对话框如图所示：



在查找内容一栏中输入需要查找的内容，可选择“全字匹配”与“区分大小写”的查找方式，在替换为一栏中输入需要替换的内容，单击“查找下一个”程序则在文档显示区域中搜索与查找内容匹配的字符串，找到第一个后则高亮显示，用户可单击“替换”将匹配的字符串替换，也可单击“全部替换”将当前文档显示区域中所有与查找内容匹配的字符串全部替换。单击“查找下一个”则继续搜索下一个匹配字符串。也可单击“取消”退出查找操作。

### 4. 保存源程序

当前运行环境下，选择菜单栏中的“文件”菜单，菜单下拉后选择“保存”，如果是无标题文档，用户需在提示下输入文档的名称及选择保存的路径，单击确定后保存；否则程序自动保存当前文档显示区域中显示的文档。或者选择菜单栏中的“文件”菜单，菜单下拉后选择“另存为”，并在提示下输入文档的名称及选择保存的路径，单击确定后保存。

## 7、编译源程序

### 编译（编译）

在当前运行环境下，选择菜单栏中的“ASM文件编译”菜单，选择编译选项则程序对当前ASM源文件进行编译，编译调试窗口中输出汇编的结果，若程序汇编有错，则详细报告错误信息。双击输出错误，集成开发环境会自动将错误所在行代码显示。

### 构建（汇编+链接）

在当前运行环境下，选择菜单栏中的“ASM文件编译”菜单，选择汇编+链接选项则程序对当前ASM源文件进行汇编与链接，编译调试窗口中输出汇编与链接的结果，若程序汇编或链接有错，则详细报告错误信息。双击输出错误，集成开发环境会自动将错误所在行代码显示。

### 重构运行（汇编+链接+执行）

在当前运行环境下，选择菜单栏中的“ASM文件编译”菜单，选择汇编+链接+执行选项则程序对当前ASM源文件执行，程序自动运行。

## 8、用户程序的调试和运行

### 1) .ASM程序的调试

#### 寄存器窗口

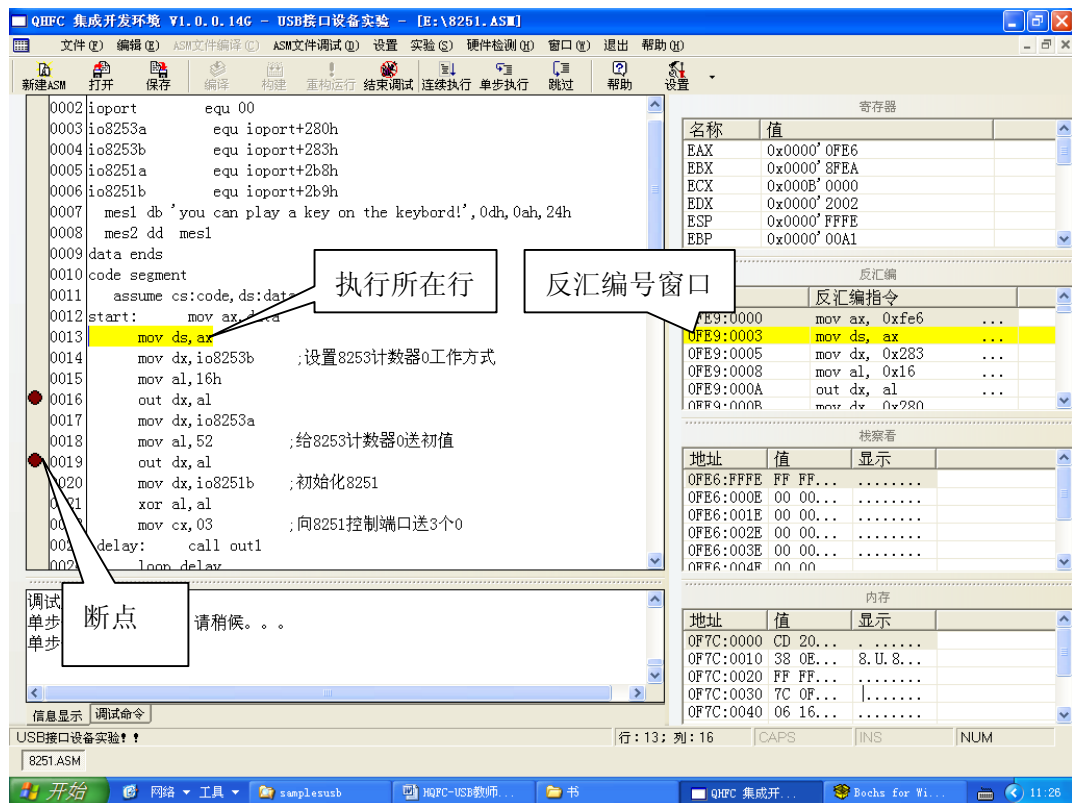
在当前运行环境下，寄存器窗口中显示主要的寄存器名称及其在当前程序中的对应值，若值为红色，即表示当前寄存器的值。调试时，单步执行，寄存器会随每次单步运行改变其输出值，同样以红色显示。

#### 开始调试

编译和链接成功之后，在“ASM文件调试”菜单中，选择“开始调试”，然也可以在工具栏中选择“开始调试”。即可开始进行程序的调试。

#### 设置/清除断点

在ASM的调试状态下，对程序代码所在某一行前最左边的灰色列条单击鼠标，即对此行前设置了断点，如果清除断点，只需再在此行前的灰色列条上的断点单击鼠标，此断点标记将被清除。箭头所指的行为当前单步执行到的所在行。设置/清除断点如图所示：



## 连续运行

在ASM的调试状态下，选择“ASM文件调试”菜单栏中的“连续运行”菜单或F5，则程序连续运行，直至碰到断点或程序运行结束。

## 单步

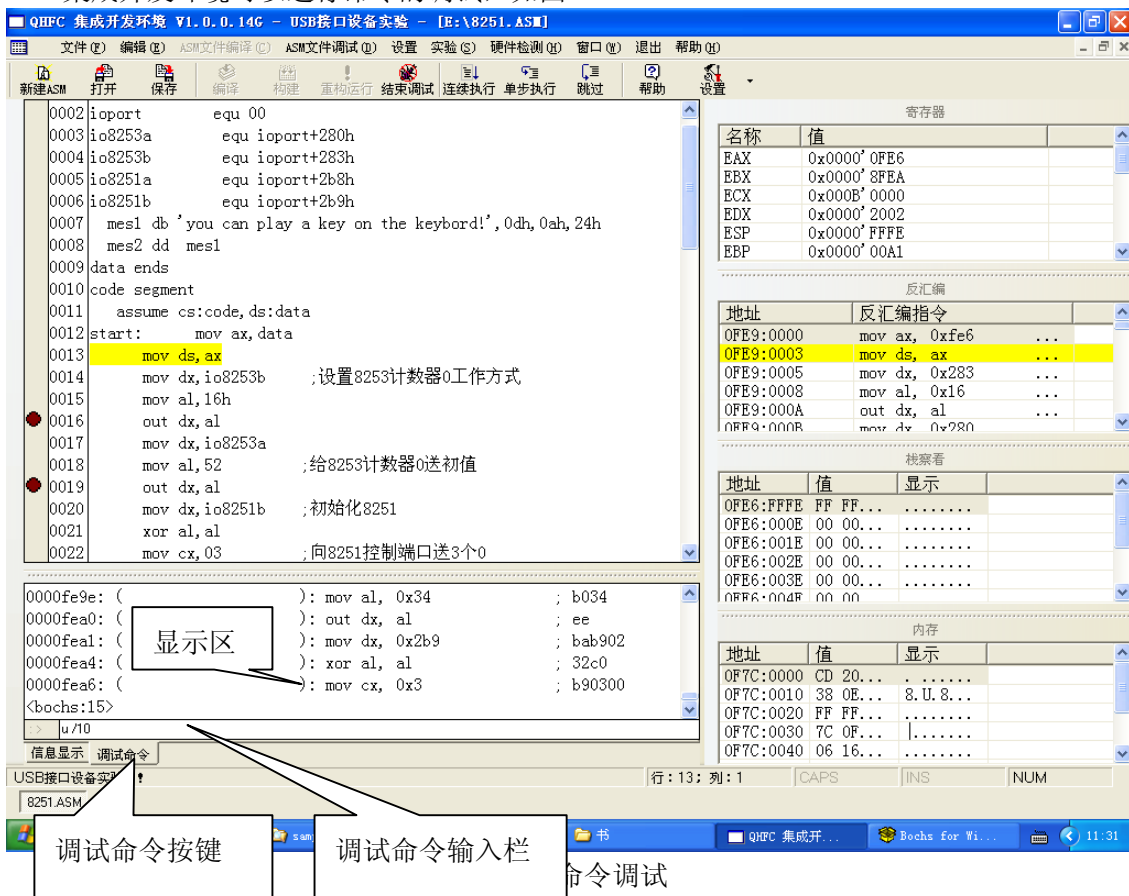
在ASM的调试状态下，选择“ASM文件调试”菜单栏中的“单步执行”菜单或F11，则程序往后运行一条语句。

## 退出调试

在ASM的调试状态下，选择“ASM文件调试”菜单栏中的“结束调试”菜单或F8，程序则退出ASM的调试状态。

## 命令调试

集成开发环境可以进行命令的调试，如图：



### 3.2.4 常用调试命令

调试指令与debug稍有区别，具体调试命令如下：

bochs提供了强大的命令行调试功能，本集成开发环境在其之上包装了一个简便易用的图形界面。如果这个界面不能满足您的要求，还可以使用命令栏直接输入调试命令与bochs交互。所有调试命令bochs都提供了简要的用法说明，输入“help”（不带引号）可查看可用的命令，help 'cmd'（带引号）可查看命令cmd相关的帮助。

下面是一些常用的命令说明及示例：

### 1. 反汇编 (u)

用法: u [/count] start end

反汇编给定的线性地址, 可选参数'count'是反汇编指令的条数

例: u 反汇编当前 cs:ip 所指向的指令

u /10 从当前 cs:ip 所指向的指令起, 反汇编10条指令

u /12 0xfeff 反汇编线性地址 0xfeff 处开始的12条指令

### 2. 查看内存 (x)

用法: x /nuf addr

查看线性地址'addr'处的内存内容

nuf 由需要显示的值个数和格式标识[xduot cbhw m]组成, 未指明用何种格式的情况下将使用上一次的格式。

x: 十六进制

c: 字符

d: 十进制

b: 字节

u: 无符号

h: 半字

o: 八进制

w: 字 (四字节)

t: 二进制

m: 使用memory dump模式

例: x /10wx 0x234 以十六进制输出位于线性地址 0x234 处的 10 个双字

x /10bc 0x234 以字符形式输出位于线性地址 0x234 处的 10 个字节

x /h 0x234 以十六进制输出线性地址 0x234 处的 1 个字

### 3. 查看寄存器 (info reg)

用法: info reg

查看CPU整数寄存器的内容

### 4. 修改寄存器 (r)

用法: r reg = expression

reg 为通用寄存器

expression 为算术表达式

例: r eax = 0x12345678 对 eax 赋值 0x12345678

r ax = 0x1234 对 ax 赋值 0x1234

r al = 0x12 + 1 对 al 赋值 0x13

### 5. 下断点 (lb)

用法: lb addr

下线性地址断点

例: lb 0xfeff 在 0xfeff 下线性地址断点, 0f00:eff 所处线性地址就是 0xfeff

### 6. 查看断点情况 (info b)

用法: info b

### 7. 删断点 (del n)

用法: del n

删除第 n 号断点

例: del 2 删除 2 号断点, 断点编号可通过前一个命令查看

## 8. 连续运行 (c)

用法: c

在未遇到断点或是 watchpoint 时将连续运行

## 9. 单步 (n 和 s)

用法: n

执行当前指令, 并停在紧接着的下一条指令。如果当前指令是 call、ret, 则相当于 Step Over。

s [count]

执行 count 条指令

## 10. 退出 (q)

用法: q

## 2. C语言程序的调试

大多数实验所用的程序需要用到配套的Visual Studio生成的静态链接库(.lib)或动态链接库(.dll)文件, 因此本软件采用了Visual C++的调试系统。由于版权问题, 本软件没有提供Visual C++的编译和调试器, 需要用户自己安装。

### 3.2 HQFC集成开发环境下VC程序的使用说明

#### 一、介绍

使用本套VC实验程序时, 请用户注意以下几点实验要求, 以便顺利完成实验:

ApiEx.dll、dll.dll、ApiExusb.h、ApiExusb.lib均为本套VC程序用到的资源。

1.在HQFC集成开环境中VC程序不需要指定所用函数的路径,直接申明后使用。

2.如果在HQFC集成开环境中VC程序申明函数时,指定了路径,请将所使用的文件放在 指定路径的目录中, 才能编译成功。

如: #include "ApiExusb.h"

把二个文件放在同VC程序同一个目录中。

新增加了三个函数:

APIEXDLL\_API bool PortWriteEx(WORD address, BYTE data, BYTE nBytesToWrite, BYTE Delay);  
APIEXDLL\_API bool PortReadEx(WORD address, BYTE buffer, BYTE nBytesToRead, BYTE nDelay);  
APIEXDLL\_API bool Read0809(WORD address, BYTE buffer, BYTE nBytesToRead, BYTE nDelay);  
其中,

address:	是读写的地址
data:	是将要写入的字节数组
buffer:	是保存读取到的数据缓冲
nBytesToWrite:	是要写的字节数
nBytesToRead:	是要读的字节数
nDelay:	是通过软件延时的因子

#### 二、函数简介

##### 1、基本输入输出-基本输入输出函数简介

1)、Startup();

语法: BOOL Startup()

功能描述: 查询PC机的微机接口实验装置是否可用, 如果可用则打开。

参数: 无

返回值：如果设备存在并且可用，则返回True，否则返回False

备注：应用程序在对板卡做任何操作之前必须调用该函数，应用程序结束时必须使用Cleanup函数关闭该设备。

2) 、void Cleanup();

语法：void Cleanup()

功能描述：关闭设备。

参数：无

返回值：无

备注：应用程序结束时必须使用Cleanup函数关闭该设备。它和Startup成对使用。

3) 、PortReadByte;

语法：BOOL PortReadByte(DWORD address, BYTE \*pdata);

功能描述：读该板卡某个的IO端口值。

参数：

address：指明要读的IO端口地址

pdata： 该函数执行完后，address所指明的端口值被填入该地址

返回值：如果读成功，则返回True，否则返回False

备注：应用程序使用该函数前必须先调用Startup函数。

例子：

```
BYTE data;
```

```
DWORD address = 0x283;
```

```
if (!Startup())
```

```
{
```

```
//ERROR... .. 出错处理
```

```
}
```

```
if(!PortReadByte(address,&data))
```

```
{
```

```
//ERROR... .. 出错处理
```

```
}
```

```
//SUCCESS... .. 成功，此时data里存放地址为address的IO端口的值
```

4) 、PortWriteByte;

语法：BOOL PortWriteByte(DWORD address, BYTE data);

功能描述：将给定值写入该板卡所指明的IO端口。

参数：

address：指明要写的硬件IO端口地址

data： 该函数执行完后，data将被写入address所指明的IO端口

返回值：如果读成功，则返回True，否则返回False

备注：应用程序使用该函数前必须先调用Startup。

例子：

```
BYTE data;
```



```

DWORD address = 0x283;
if (!Startup())
{
//ERROR... .. 出错处理
}
if (!PortReadByte(address, &data))
{
//ERROR... .. 出错处理
}
//SUCCESS... .. 此时已经将值data写入address所指明的IO端口

```

## 2、中断-中断函数简介

### 1)、EnableIntr;

语法: BOOL EnableIntr();

功能描述: 将微机实验装置的中断输入设为有效, 执行此函数后, 将接受微机实验装置上的中断请求, 然后根据该请求申请一个PCI中断。

参数: 无

返回值: 如果成功, 则返回True, 否则返回False

备注: 应用程序在调用该函数之前, 必须先调用Startup函数。

### 2)、DisableIntr;

语法: BOOL DisableIntr();

功能描述: 将微机实验装置的中断输入设为无效, 执行此函数后, 将不响应微机实验装置上的中断请求

参数: 无

返回值: 如果成功, 则返回True, 否则返回False

备注: 应用程序在调用该函数之前, 必须先调用Startup函数。

### 3)、RegisterLocalISR;

语法: BOOL RegisterLocalISR (ISR\_ROUTINE pfuncISR, data);

功能描述: 注册中断服务程序, 当微机实验箱上的中断输入有效时, 且实验箱上的中断输入使能, 程序将会执行该中断服务程序。

参数: pfuncISR: 该参数即为中断服务函数名

返回值: 如果成功, 则返回True, 否则返回False

参数: data: 该参数即为中断号。

备注: 应用程序在调用该函数之前, 必须先调用Startup函数。

## 3、DMA函数简介

### 1)、Write8237

语法: bool Write8237(WORD address, BYTE data);

功能描述: 写USB核心板上8237的某个端口。

参数: address: 指明要写的8237的端口地址

Data: 该函数执行完后, data将被写入address所指明的8237端口。

返回值： 如果写成功，返回True, 否则返回False。

备注：使用该函数前必须先调用 Startup函数。

#### 2)、Read8237

语法：bool Read8237(WORD address, BYTE\* pdata);

功能描述：读USB核心板上8237的某个端口。

参数： address： 指明要读的8237的端口地址

pdata： 该函数执行完后，address所指明的8237端口值被填入该地址。

返回值： 如果写成功，返回True, 否则返回False。

备注：使用该函数前必须先调用 Startup函数。

### 4、存储器读写-设备存储器读写函数简介

#### 1)、MemReadByte;

语法：BOOL MemReadByte(DWORD address, BYTE \*pdata);

功能描述：读该板卡某个映射的存储器地址的值。

参数：

address： 指明要读的存储器地址

pdata： 该函数执行完后，address所指明的存储器地址的值被填入该地址

返回值：如果读成功，则返回True，否则返回False

备注：应用程序使用该函数前必须先调用Startup函数。

例子：

BYTE data;

DWORD address = 0x283;

if (!Startup())

{

//ERROR... .. 出错处理

}

if (!MemReadByte(address, &data))

{

//ERROR... .. 出错处理

}

//SUCCESS... .. 成功，此时data里存放地址为address的存储器地址的值

#### 2)、MemWriteByte;

语法：BOOL MemWriteByte(DWORD address, BYTE data);

功能描述：将给定值写入该板卡所指明的存储器地址的值。

参数：

address： 指明要写的硬件存储器地址

data： 该函数执行完后，data将被写入address所指明的存储器地址

返回值：如果读成功，则返回True，否则返回False

备注：应用程序使用该函数前必须先调用Startup。

例子：

```

BYTE data;
DWORD address = 0x283;
if (!Startup())
{
//ERROR... .. 出错处理
}
if (!MemReadByte(address, &data))
{
//ERROR... .. 出错处理
} //SUCCESS... .. 此时已经将值data写入address所指明的存储器地址

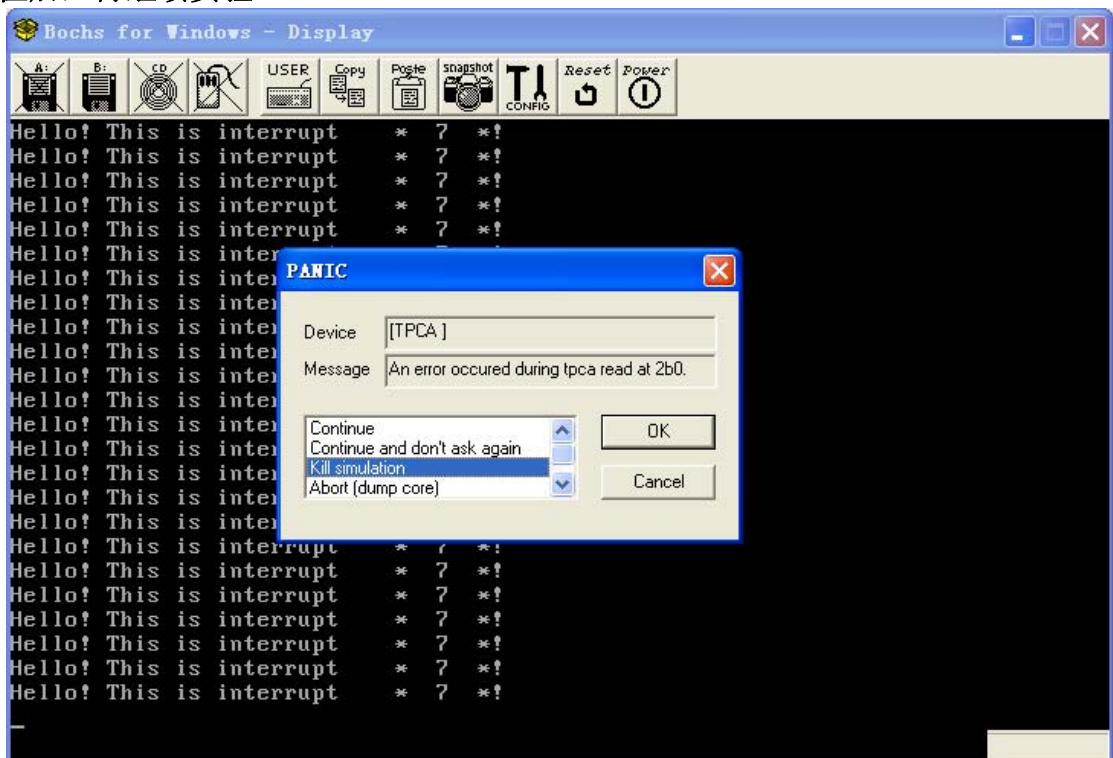
```

### C语言程序的调试

大多数实验所用的程序需要用到配套的VisualStudio生成的静态链接库(.lib)或动态链接库(.dll)文件，因此本软件采用了Visual C++的系统。由于版权问题，本软件没有提供Visual C++的调试器，需要用户自己安装。

### 常见问题提示：

该实验台为USB接口，在实验中需要频频接触实验台。因为人体带电和其它原因，容易造成通信干扰，使其设备通信中断。出现如下现象。出现该现象时请按USB接口核心小板上的复位按键或关闭大板电源再重新打开。使硬件通信复位后，再继续实验。



## 第四章 基本实验

### 一、几点约定：

1、实验电路介绍中凡不加“利用通用插座”说明的均为实验台上已固定电路。

### 实验一 I/O地址译码

#### 一、实验目的

掌握I/O地址译码电路的工作原理。

#### 二、实验原理和内容

1、实验电路如图1-1所示，其中74LS74为D触发器，可直接使用实验台上数字电路实验区的D触发器，74LS138为地址译码器。译码输出端Y0~Y7在实验台上“I/O地址”输出端引出，每个输出端包含8个地址，Y0：280H~287H，Y1：288H~28FH，…… 当CPU执行I/O指令且地址在280H~2BFH范围内，译码器选中，必有一根译码线输出负脉冲。

例如：执行下面两条指令

MOV DX, 2A0H

OUT DX, AL (或IN AL, DX)

Y4输出一个负脉冲，执行下面两条指令

MOV DX, 2A8H

OUT DX, AL (或IN AL, DX)

Y5输出一个负脉冲。

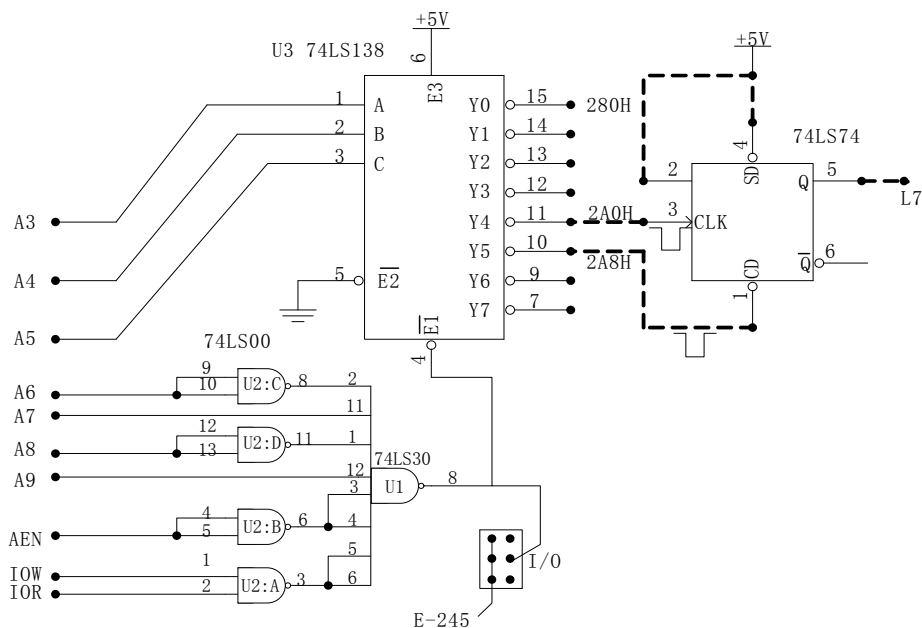


图4-1-1

利用这个负脉冲控制L7闪烁发光（亮、灭、亮、灭、……），时间间隔通过软件延时实现。

2、接线：	Y4/IO地址	接	CLK/D触发器	
	Y5/IO地址	接	CD/D触发器	
	D/D触发器	接	SD/D角发器	接 +5V
	Q/D触发器	接	逻辑笔	

### 三、编程提示

1、实验电路中D触发器CLK端输入脉冲时，上升沿使Q端输出高电平L7发光，CD端加低电平L7灭。

2、参考程序：YMQ.ASM

```

output1      equ 2a0h
output2      equ 2a8h
code segment
    assume cs:code
start:
    mov dx,output1
    out dx,al
    call delay      ;调延时子程序
    mov dx,output2
    out dx,al
    call delay      ;调延时子程序
    mov ah,1
    int 16h
    je start
    mov ah,4ch
    int 21h
delay proc near      ;延时子程序
    mov bx,200
111:    mov cx,0
11:     loop 11
        dec bx
        jne 111
        ret
delay endp
code ends
end start

```

## 实验二 简单并行接口

### 一、实验目的

掌握简单并行接口的工作原理及使用方法。

### 二、实验原理和内容

1、按下面图4-2-1简单并行输出接口电路图连接线路(74LS273插通用插座, 74LS32用实验台上的“或门”)。74LS273为8D触发器, 8个D输入端分别接数据总线D0~D7, 8个Q输出端接LED显示电路L0~L7。

2、编程从键盘输入一个字符或数字, 将其ASC II 码通过这个输出接口输出, 根据8个发光二极管发光情况验证正确性。

3、按下面图4-2-2简单并行输入接口电路图连接电路(74LS244插通用插座, 74LS32用实验台上的“或门”)。74LS244为八缓冲器, 8个数据输入端分别接逻辑电平开关输出K0~K7, 8个数据输出端分别接数据总线D0~D7。

4、用逻辑电平开关预置某个字母的ASC II 码, 编程输入这个ASC II 码, 并将其对应字母在屏幕上显示出来。

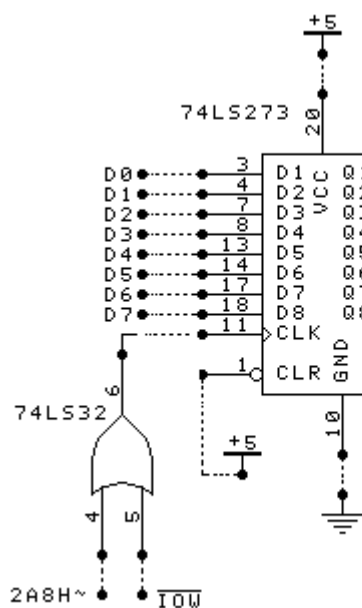


图4-2-1

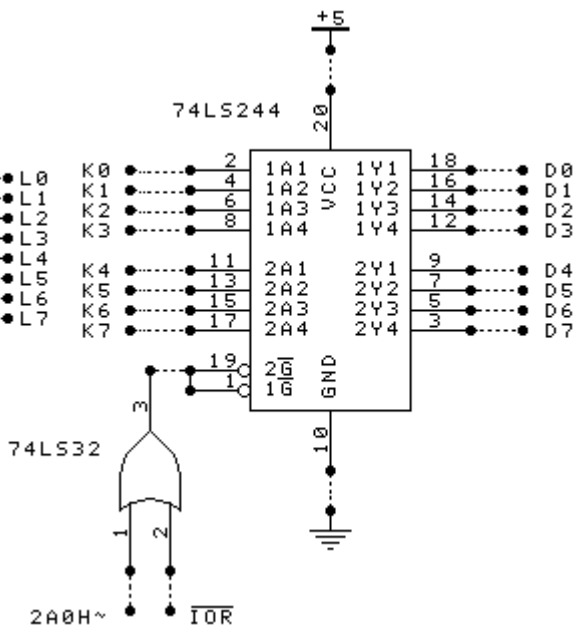


图4-2-2

5、接线: 1) 输出

按图3-3-1接线 (图中虚线为实验所需接线, 74LS32为实验台逻辑或门)

2) 输入

按图3-3-2接线 (图中虚线为实验所需接线, 74LS32为实验台逻辑或门)

### 三、编程提示

1、上述并行输出接口的地址为2A8H, 并行输入接口的地址为2A0H, 通过上述并行接口电路输出数据需要3条指令:

```

MOV  AL, 数据
MOV  DX, 2A8H
OUT  DX, AL

```

通过上述并行接口输入数据需要2条指令：

```

MOV  DX, 2ADH
IN   AL, DX

```

## 2、参考流程图

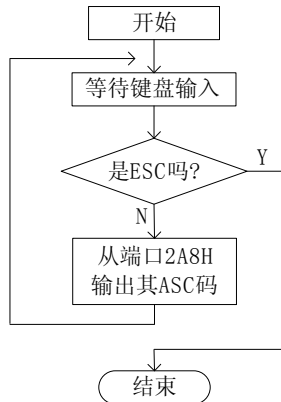


图2-3 参考程序1

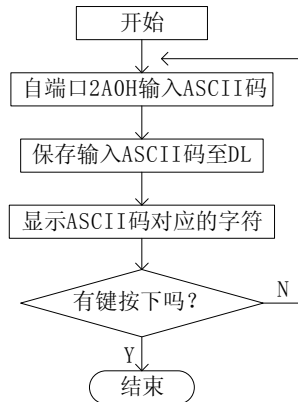


图2-4 参考程序2

## 3、参考程序1:

```

OUT. ASM
ls273  equ 2a8h
code segment
assume cs:code
start:  mov ah, 2          ;回车符
        mov dl, 0dh
        int 21h
        mov ah, 1          ;等待键盘输入
        int 21h
        cmp al, 27         ;判断是否为ESC键
        je exit            ;若是则退出
        mov dx, ls273      ;若不是, 从2A8H输出其ASCII码
        out dx, al
        jmp start          ;转start
exit:   mov ah, 4ch        ;返回
        int 21h
code ends
end start

```

## 4、参考程序2:

```

IN. ASM
ls244    equ 2a0h
code segment
assume cs:code
start:
    mov dx, ls244    ;从2A0输入一数据
    in al, dx
    mov dl, al        ;将所读数据保存在DL中
    mov ah, 02
    int 21h
    mov dl, 0dh        ;显示回车符
    int 21h
    mov dl, 0ah        ;显示换行符
    int 21h
    mov ah, 06        ;是否有键按下
    mov dl, 0ffh
    int 21h
    jnz exit
    je start          ;若无, 则转start
exit:    mov ah, 4ch    ;返回
    int 21h
code ends
end start

```



### 实验三 可编程并行接口8255

#### 一、实验目的

1、通过实验，掌握8255工作于方式0以及设置A口为输出口，C口为输入口的方法。

#### 二、实验原理和内容

- 1、实验电路如图4-3-1，8255C口接逻辑电平开关K0~K7，A口接LED显示电路L0~L7。
- 2、编程从8255C口输入数据，再从A口输出。

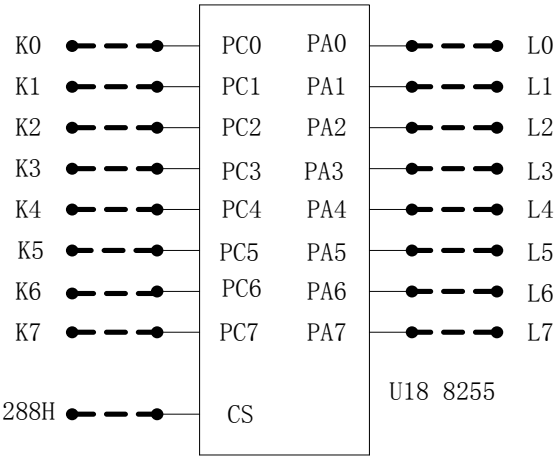


图4-3-1

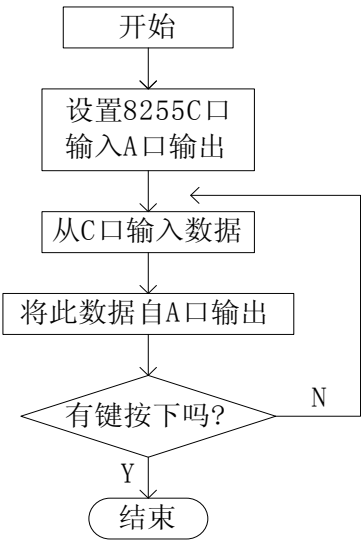


图4-3-2

- 3、接线：
- |              |   |              |
|--------------|---|--------------|
| PC7~PC0/8255 | 接 | K7~K0/逻辑电平开关 |
| PA7~PA0/8255 | 接 | L7~L0/LED显示  |
| CS/8255      | 接 | Y1/IO地址      |

#### 三、编程提示

1、8255控制寄存器端口地址--28BH， A口的地址--288H， C口的地址--28AH

2、参考流程图(图4-3-2)

3、参考程序1: E8255-I0.ASM

```
io8255a equ 288h
io8255b equ 28bh
io8255c equ 28ah
code segment
assume cs:code
```

```

start:  mov dx, io8255b           ; 设8255为C口输入, A口输出
        mov al, 8bh
        out dx, al
inout:  mov dx, io8255c           ; 从C口输入一数据
        in al, dx
        mov dx, io8255a           ; 从A口输出刚才自C口
        out dx, al               ; 所输入的数据
        mov dl, 0ffh             ; 判断是否有按键
        mov ah, 06h
        int 21h
        jz inout                 ; 若无, 则继续自C口输入, A口输出
        mov ah, 4ch              ; 否则返回
        int 21h
code ends
end start

```

## 实验四 七段数码管

### 一、实验目的

掌握数码管显示数字的原理

### 二、实验原理和内容

- 1、静态显示:按4-4-1连接好电路,将8255的A口PA0~PA7分别与七段数码管的段码驱动输入端a~dp相连,位码驱动输入端S0、S1接PC0、PC1; S2、S3接地。编程在数码管上循环显示“00-99”。

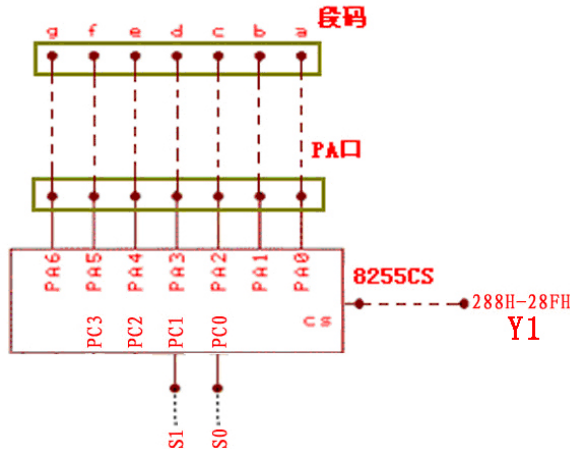


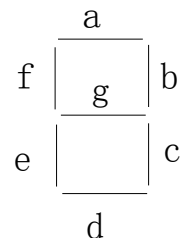
图4-4-1

- |                    |   |               |
|--------------------|---|---------------|
| 2、接线: PA7~PA0/8255 | 接 | dp~a /LED数码管  |
| PC1~PC0/8255       | 接 | S1~S0 /LED数码管 |
| GND                | 接 | S3~S2 /LED数码管 |
| CS/8255            | 接 | Y1 /IO地址      |

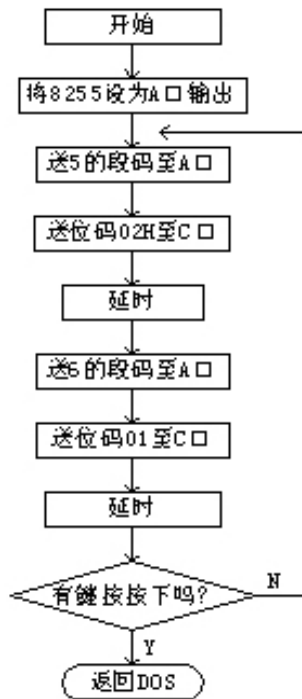
### 三、编程提示

- 1、实验台上的七段数码管为共阴型,段码采用同相驱动,输入端加高电平,选中的数码管亮,位码加反相驱动器,位码输入端高电平选中。七段数码管的字型代码表如下表:

显示字形	g	e	f	d	c	b	a	段码
0	0	1	1	1	1	1	1	3fh
1	0	0	0	0	1	1	0	06h
2	1	0	1	1	0	1	1	5bh
3	1	0	0	1	1	1	1	4fh
4	1	1	0	0	1	1	0	66h
5	1	1	0	1	1	0	1	6dh
6	1	1	1	1	1	0	1	7dh
7	0	0	0	0	1	1	1	07h
8	1	1	1	1	1	1	1	7fh
9	1	1	0	1	1	1	1	6fh



## 2、参考流程图



## 3、参考程序1: LED.ASM

```

data segment
io8255a equ 28ah
io8255b equ 28bh
io8255c equ 288h
led      db 3fh, 06h, 5bh, 4fh, 66h, 6dh, 7dh, 07h, 7fh, 6fh ;段码
buffer1  db 0, 0 ;存放要显示的十位和个位
bz       dw ? ;位码
data ends

code segment
assume cs:code, ds:data
start:
    mov ax, data
    mov ds, ax
    mov dx, io8255b ;将8255设为A口输出
    mov al, 80h
    out dx, al
    mov di, offset buffer1 ;设di为显示缓冲区
loop1: mov cx, 030h ;循环次数

```

```

loop2:  mov bh,02
l11:    mov byte ptr bz,bh
        push di
        dec di
        add di, bz
        mov bl,[di]           ;bl为要显示的数
        pop di
        mov bh,0
        mov si,offset led     ;置led数码表偏移地址为SI
        add si,bx             ;求出对应的led数码
        mov al,byte ptr [si]
        mov dx,io8255c        ;自8255A的口输出
        out dx,al
        mov al,byte ptr bz    ;使相应的数码管亮
        mov dx,io8255a
        out dx,al
        push cx
        mov cx,100
delay:  loop delay            ;延时
        pop cx
        mov al,00h
        out dx,al
        mov bh,byte ptr bz
        shr bh,1
        jnz l11
        loop loop2           ;循环延时
        mov ax,word ptr [di]
        cmp ah,09
        jnz set
        cmp al,09
        jnz set
        mov ax,0000
        mov [di],al
        mov [di+1],ah
        jmp loop1
set:    mov ah,01
        int 16h
        jne exit             ;有键按下则转exit
        mov ax,word ptr [di]

```

```

        inc al
        aaa
        mov [di], al           ;al为十位
        mov [di+1], ah        ;ah中为个位
        jmp loop1
exit:    mov dx, io8255a
        mov al, 0              ;关掉数码管显示
        out dx, al
        mov ah, 4ch            ;返回
        int 21h
code ends
end start

```

## 实验五 键盘显示控制实验

### 一、实验目的

- 1、掌握8255控制键盘及显示电路的基本功能及编程方法。
- 2、掌握一般键盘和显示电路的工作原理。

### 二、实验内容

- 1、编程：使得在小键盘上每按一个键，4位数码管上显示出相应字符，它们的对应关系如下：

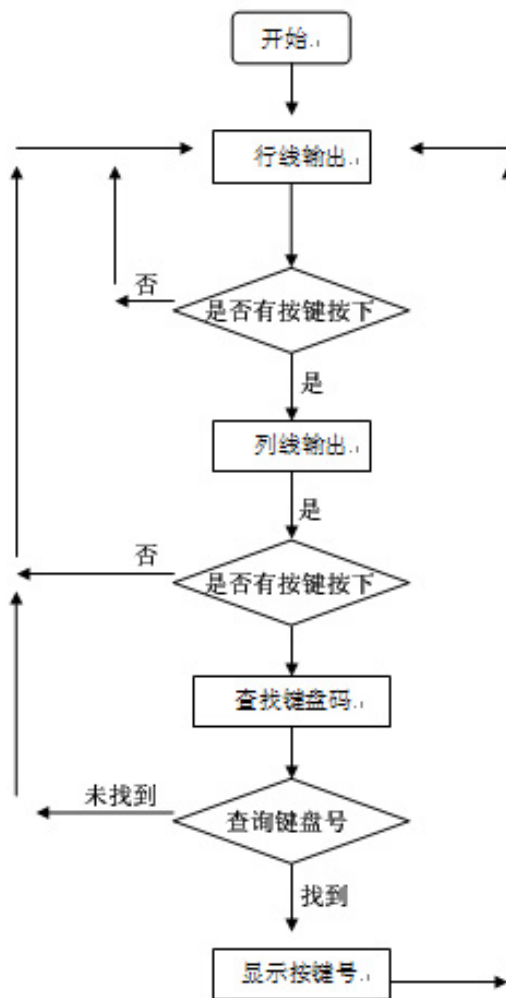
小键盘		显示	小键盘		显示
0	—	0	C	—	C
1	—	1	D	—	d
2	—	2	E	—	E
3	—	3	F	—	F
4	—	4			
5	—	5			
6	—	6			
7	—	7			
8	—	8			
9	—	9			
A	—	日			
B	—	b			

- 2、接线：
- |               |   |               |
|---------------|---|---------------|
| PC7~PC0 /8255 | 接 | 行3~列0 /4X4键盘  |
| PA7~PA0 /8255 | 接 | dp~a /LED数码管  |
| CS/8255       | 接 | Y1 /IO地址      |
| +5V           | 接 | S0 /LED数码管    |
| GND           | 接 | S3~S1 /LED数码管 |

### 三、编程提示

设置8255 C 口键盘输入、A 口为数码管段码输出。

### 四、参考流程图



## 五、参考程序

;386以上微机适用

;tasm4.1或以上编译

;\*\*\*\*\*;

;\* 键盘显示 8255LED \*;

;\*\*\*\*\*;

;\*\*\*\*\*;

;\* 8255薄膜按键实验 \*;

;\*\*\*\*\*;

a8255 equ 288H ;8255 A口

c8255 equ 28aH ;8255 C口

k8255 equ 28bH ;8255控制口

data segment

table1 dw 0770h,0B70h,0D70h,0E70h,07B0h,0BB0h,0DB0h,0EB0h



```

        dw 07D0h, 0BD0h, 0DD0h, 0ED0h, 07E0h, 0BE0h, 0DE0h, 0EE0h      ;键盘
扫描码表
LED      DB 3FH, 06H, 5BH, 4FH, 66H, 6DH, 7DH, 07H, 7FH, 6FH, 77H, 7CH
        DB 39h, 5EH, 79h, 71h, 0ffh      ;LED段码表,
0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f
char     db '0123456789ABCDEF'          ;字符表
mes      db 0ah, 0dh, 'PLAY ANY KEY IN THE SMALL KEYBOARD! ', 0ah, 0dh
        db 'IT WILL BE ON THE SCREEN! END WITH E ', 0ah, 0dh, '$'
key_in   db 0h
data     ends
stacks   segment stack      ;堆栈空间
        db 100 dup (?)
stacks   ends
code     segment
        assume cs:code, ds:data, ss:stacks, es:data
start:
        cli
        mov ax, data
        mov ds, ax
        mov es, ax
        mov ax, stacks
        mov ss, ax
        mov dx, offset mes      ;显示提示信息
        mov ah, 09
        int 21h
        MOV DX, k8255           ;初始化8255控制字
        mov al, 81h
        out dx, al
main_key:
        call key                ;get a char in (key_in) and display it
        call disply             ;调显示子程序, 显示得到的字符
        cmp byte ptr key_in, 'E'
        jnz main_key
        mov ax, 4c00h           ;if (dl)='E' return to EXIT!
        int 21h                 ;退出
key proc near
key_loop:
        mov ah, 1
        int 16h

```

```

    jnz exit                                ;pc键盘有键按下则退出

    mov dx, c8255
    mov al, 0fh
    out dx, al
    in al, dx                              ;读行扫描值
    and al, 0fh
    cmp al, 0fh
    jz key_loop                            ;未发现有键按下则转
    call delay                             ;delay for amoment
    mov ah, al
    MOV DX, k8255
    mov al, 88h
    out dx, al
    mov dx, c8255
    mov al, ah
    or al, 0f0h
    out dx, al
    in al, dx                              ;读列扫描值
    and al, 0f0h
    cmp al, 0f0h
    jz key_loop                            ;未发现有键按下则转

    mov si, offset table1                  ;键盘扫描码表首址
    mov di, offset char                    ;字符表首址
    mov cx, 16                             ;待查表的表大小
key_tonext:
    cmp ax, [si]                          ;cmp (col,row) with every word
    jz key_findkey                         ;in the table
    dec cx
    jz key_loop                            ;未找到对应扫描码
    add si, 2
    inc di
    jmp key_tonext
key_findkey:
    mov dl, [di]
    mov ah, 02
    int 21h                               ;显示查找到的键盘码
    mov byte ptr key_in, dl

```

```

key_waitup:
    MOV DX,k8255
    mov al,81h
    out dx,al
    mov dx,c8255
    mov al,0fh
    out dx,al
    in al,dx          ;读行扫描值
    and al,0fh
    cmp al,0fh
    jnz key_waitup    ;按键未抬起转
    call delay         ;delay for amoment
    ret
exit:      mov byte ptr key_in,'E'
    ret
key endp
delay proc near
    push ax            ;delay 50ms--100ms
    mov ah,0
    int 1ah
    mov bx,dx
delay1:
    mov ah,0
    int 1ah
    cmp bx,dx
    jz delay1
    mov bx,dx
delay2:
    mov ah,0
    int 1ah
    cmp bx,dx
    jz delay2
    pop ax
    ret
delay endp
DISPLY    PROC NEAR
    PUSH ax
    MOV BX,OFFSET LED
    MOV AL,byte ptr key_in

```

```

        SUB al, 30h
        CMP al, 09h
        JNG  DIS2
        SUB al, 07h
DIS2:   XLAT
        MOV DX, a8255
        OUT DX, AL           ;输出显示数据，段码
        POP AX
        RET
DISPLY   ENDP

code ends
end start

```

## 实验六 竞赛抢答器

### 一、实验目的

- 1、了解微机化竞赛抢答器的基本原理。
- 2、进一步学习使用并行接口。

### 二、实验原理和内容

1、图4—6-1为竞赛抢答器（模拟）的原理图，逻辑开关K0~K7代表竞赛抢答按钮0~7号，当某个逻辑电平开关置“1”时，相当某组抢答按钮按下。在七段数码管上将其组号（0~7）显示出来。

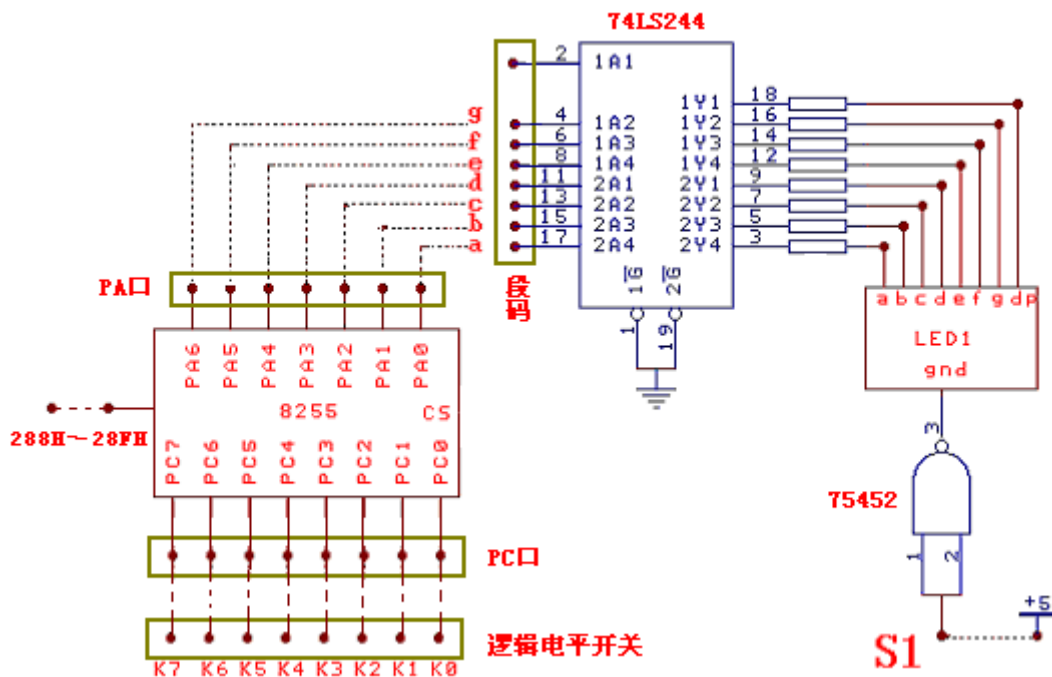


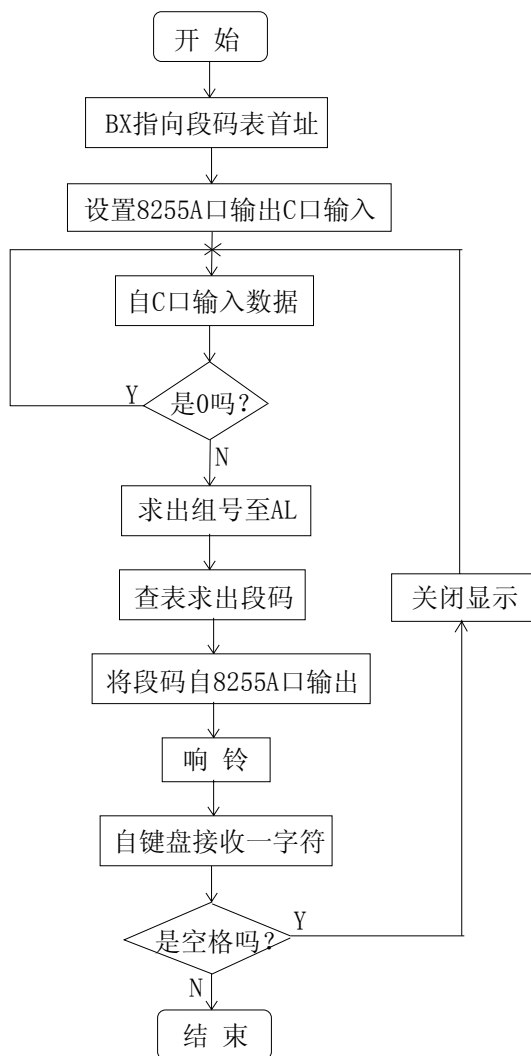
图 4-6-1

2、接线：	PC7~PC0 /8255	接	K7~K0 /逻辑电平开关
	PA7~PA0 /8255	接	dp~a /LED数码管
	CS /8255	接	Y1 /IO地址
	GND	接	S3、S2、S0 /LED数码管
	+5V	接	S1 /LED数码管

### 三、编程提示

设置8255为C口输入、A口输出，读取C口数据，若为0表示无人抢答，若不为0则有人抢答。根据读取数据可判断其组号。从键盘上按空格键开始下一轮抢答，按其它键程序退出。

### 四、参考流程图



## 五、参考程序

### 1、参考程序： QDQ.ASM

```

data segment
ioport      equ 0e800h-0280h
io8255a     equ ioport+28ah
io8255b     equ ioport+28bh
io8255c     equ ioport+288h
led         db      3fh, 06h, 5bh, 4fh, 66h, 6dh, 7dh, 07h ;数码表
data ends

code segment
    assume cs:code, ds:data
start:mov ax, data
    mov ds, ax
    mov dx, io8255b      ;设8255为A口输出, C口输入

```

```

    mov ax, 89h
    out dx, al
    mov bx, offset led ;使BX指向段码管首址
sss:   mov dx, io8255a
        in  al, dx      ;从8255的C口输入数据
        or  al, al      ;比较是否为0
        je  sss        ;若为0,则表明无键按下, 转sss
        mov cl, 0ffh    ;cl作计数器, 初值为-1
rr:    shr al, 1
        inc cl
        jnc rr
        mov al, cl
        xlat
        mov dx, io8255c
        out dx, al
        mov dl, 7       ;响铃 ASCII码为07
        mov ah, 2
        int 21h
wai:   mov ah, 1
        int 21h
        cmp al, 20h     ;是否为空格
        jne eee        ;不是, 转eee
        mov al, 0       ;是, 关灭灯
        mov dx, io8255c
        out dx, al
        jmp sss
eee:   mov ah, 4ch      ;返回
        int 21h
code  ends
      end start

```

## 实验七 交通灯控制实验

### 一. 实验目的

通过并行接口8255实现十字路口交通灯的模拟控制, 进一步掌握对并行口的使用。

### 二. 实验原理和内容

1、如图4-7-1, L7、L6、L5作为南北路口的交通灯与PC7、PC6、PC5相连, L2、L1、L0作为东西路口的交通灯与PC2、PC1、PC0相连。编程使六个灯按交通灯变化规律亮灭。

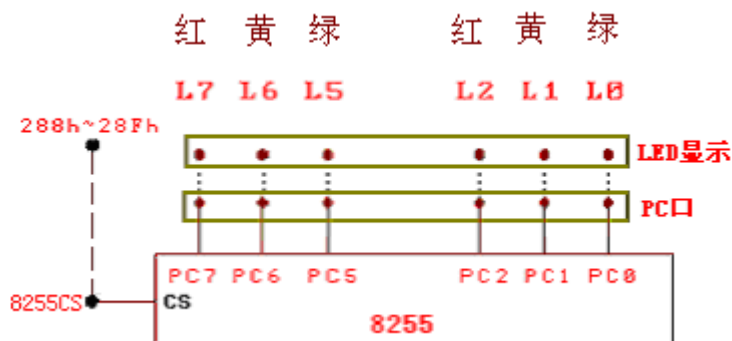


图 4-7-1

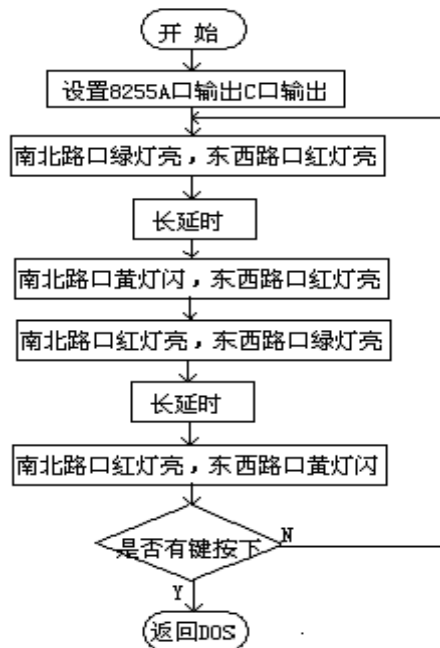
2、接线： PC7~PC0 /8255 接 L7~L0 /LED显示  
CS /8255 接 Y1 /IO地址

### 三. 编程提示：十字路口交通灯的变化规律要求：

- (1) 南北路口的绿灯、东西路口的红灯同时亮30秒左右。
- (2) 南北路口的黄灯闪烁若干次，同时东西路口的红灯继续亮。
- (3) 南北路口的红灯、东西路口的绿灯同时亮30秒左右。
- (4) 南北路口的红灯继续亮、同时东西路口的黄灯亮闪烁若干次。
- (5) 转 (1) 重复。

### 四、参考流程图





## 五、参考程序

### 1、参考程序： JTD.ASM

```

;*****
;*  十字路口红绿灯模拟演示程序  *;
;*  端口各灯的设置:              *;
;*  1红 1黄 1绿 0 0 2红 2黄 2绿  *;
;*****
data segment
ioport      equ 0e800h-0280h
io8255a      equ ioport+28ah
io8255b      equ ioport+28bh
portc1  db  24h, 44h, 04h, 44h, 04h, 44h, 04h    ;六个灯可能
          db  81h, 82h, 80h, 82h, 80h, 82h, 80h    ;的状态数据
          db  0ffh                                ;结束标志
data ends
code  segment
      assume  cs:code, ds:data
start:
      mov  ax, data
      mov  ds, ax
      mov  dx, io8255b
      mov  al, 90h
      out  dx, al          ;设置8255为C口输出

```

```

        mov     dx, io8255a
re_on:   mov     bx, 0
on:      mov     al, portc1[bx]
        cmp     al, 0ffh
        jz      re_on
        out     dx, al           ;点亮相应的灯
        inc     bx
        mov     cx, 20          ;参数赋初值
        test    al, 21h         ;是否有绿灯亮
        jz      del            ;没有, 短延时
        mov     cx, 5000        ;有, 长延时
del:     mov     di, 9000        ;di赋初值5000
de0:     dec     di              ;减1计数
        jnz     de0             ;di不为0
        loop    del
        push    dx
        mov     ah, 06h
        mov     dl, 0ffh
        int     21h
        pop     dx
        jz      on              ;没有, 转到on
exit:    mov     ah, 4ch         ;返回
        int     21h
code ends
        end     start

```

## 实验八 可编程定时器 / 计数器（8254）

### 一、实验目的

掌握8254的基本工作原理和编程方法，用示波器观察不同方式下的波形。

### 二、实验原理和内容

- 1、按图4-8-1虚线连接电路，将计数器0设置为方式0，计数器初值为N( $N \leq 0FH$ )，用手动逐个输入单脉冲，编程使计数值在屏幕上显示，并同时用逻辑笔观察OUT0电平变化(当输入N+1个脉冲后OUT0变高电平)。

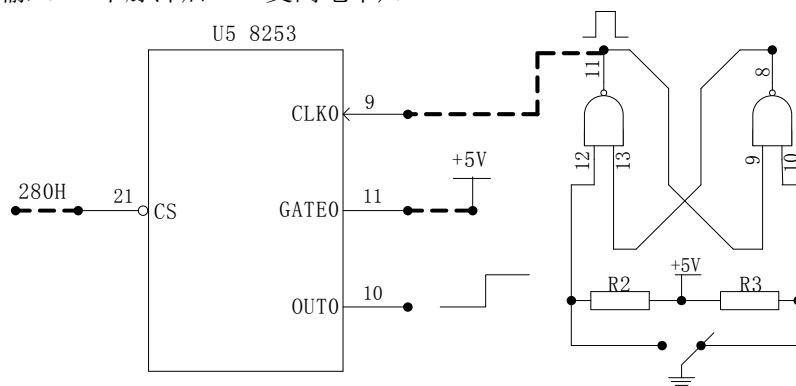


图4-8-1

- 2、按图3-2连接电路，将计数器0、计数器1分别设置为方式3，计数初值设为1000，用逻辑笔观察OUT1输出电平的变化(频率1HZ)。

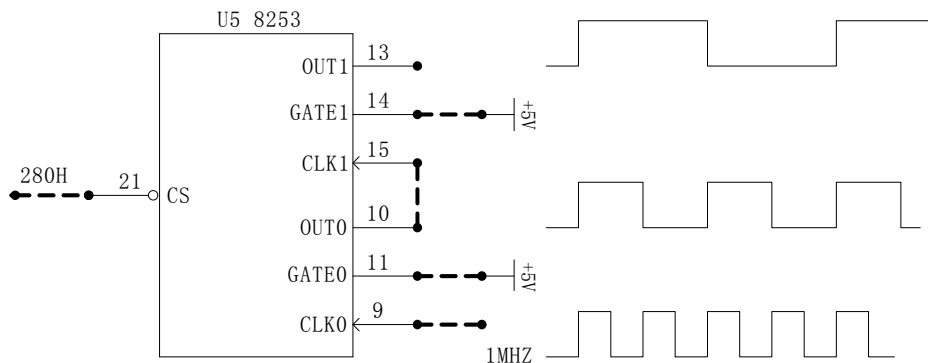


图4-8-2

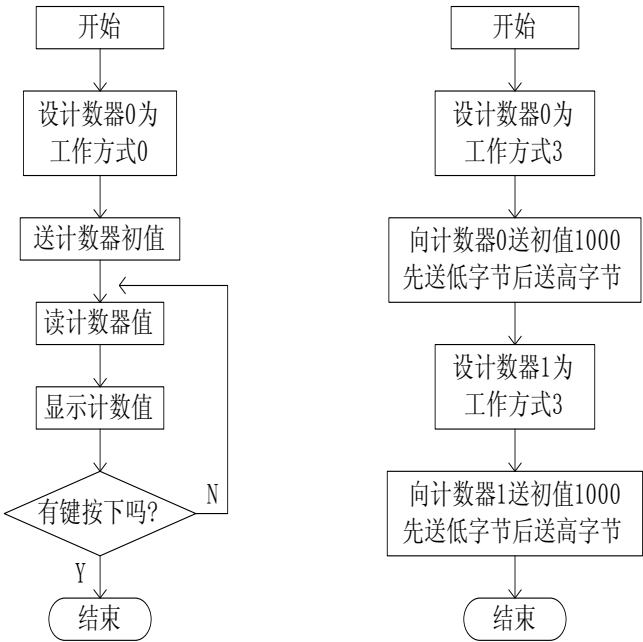
- 3、接线：
 

1)、CS /8254	接	Y0 /IO 地址
GATE0 /8254	接	+5V
CLK0 /8254	接	单脉冲
2)、CS /8254	接	Y0 /IO 地址
GATE0 /8254	接	+5V
CLK0 /8254	接	1M时钟
OUT0 /8254	接	CLK1 /8254
GATE1 /8254	接	+5V

三、编程提示

1、8254控制寄存器地址	283H
计数器0地址	280H
计数器1地址	281H
CLK0连接时钟	1MHZ

2、参考流程图



四、参考程序 1、参考程序1: JSQ.ASM

```
io8254a equ 283h
io8254b equ 280h
code segment
assume cs:code
start:
    mov al, 14h ;设置8254通道0为工作方式2, 二进制计数
    mov dx, io8254a
    out dx, al
    mov dx, io8254b ;送计数初值为0FH
    mov al, 0fh
    out dx, al
l11: in al, dx ;读计数初值
    call disp ;调显示子程序
    push dx
```

```

        mov ah, 06h
        mov dl, 0ffh
        int 21h
        pop dx
        jz l1l
        mov ah, 4ch           ;退出
        int 21h
disp proc near                ;显示子程序
        push dx
        and al, 0fh           ;首先取低四位
        mov dl, al
        cmp dl, 9             ;判断是否<=9
        jle num               ;若是则为'0'-'9', ASCII码加30H
        add dl, 7             ;否则为'A'-'F', ASCII码加37H
num:    add dl, 30h
        mov ah, 02h           ;显示
        int 21h
        mov dl, 0dh           ;加回车符
        int 21h
        mov dl, 0ah           ;加换行符
        int 21h
        pop dx
        ret                   ;子程序返回
disp endp
code ends
end start

```

## 2、参考程序2: DSQ.ASM

```

io8254a    equ 280h
io8254b    equ 281h
io8254c    equ 283h
code segment
assume     cs:code
start:
        mov dx, io8254c       ;向8254写控制字
        mov al, 36h           ;使0通道为工作方式3

```

```

    out dx,al
    mov ax,1000          ;写入循环计数初值1000
    mov dx,io8254a
    out dx,al            ;先写入低字节
    mov al,ah
    out dx,al            ;后写入高字节
    mov dx,io8254c
    mov al,76h           ;设8254通道1工作方式2
    out dx,al
    mov ax,1000          ;写入循环计数初值1000
    mov dx,io8254b
    out dx,al            ;先写低字节
    mov al,ah
    out dx,al            ;后写高字节
    mov ah,4ch           ;程序退出
    int 21h
code ends
end start

```

## 实验九 继电器控制

### 一、实验目的

- 1、了解微机控制直流继电器的一般方法。
- 2、进一步熟悉使用8255、8254。

### 二、实验原理和内容

1、实验电路如图4-9-1，按虚线连接电路：CLK0接1MHz，GATE0，GATE1，接+5V，OUT0接CLK1，OUT1接PA0，PC0接继电器驱动电路的开关输入端Ik。继电器常开触点串联一个发光二极管，编程使用8254定时，让继电器周而复始的闭合5秒钟(指示灯灯亮)，断开5秒钟(指示灯灯灭)。

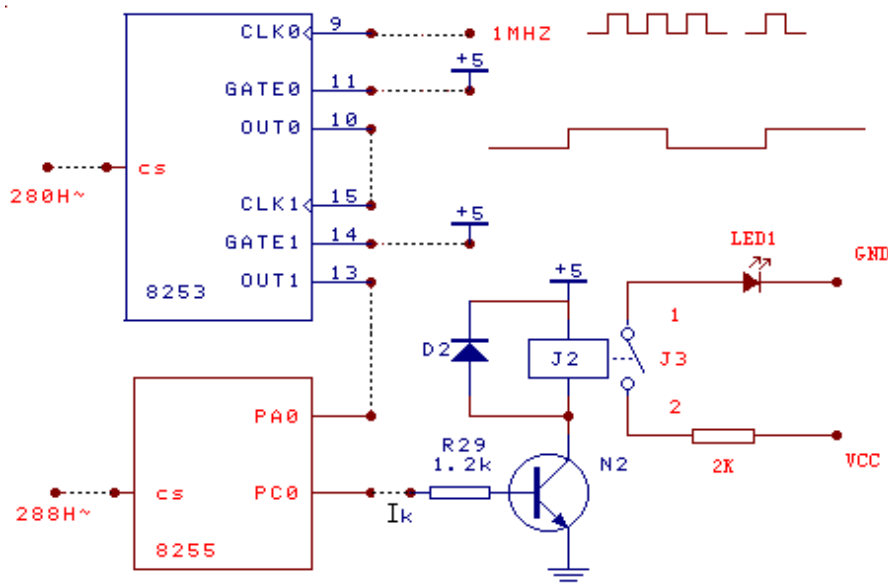


图4-9-1

2、接线：	CS /8254	接	Y0 /IO 地址
	GATE0 /8254	接	+5V
	CLK0 /8254	接	1M时钟
	OUT0 /8254	接	CLK1 /8254
	GATE1 /8254	接	+5V
	OUT1 /8254	接	PC7 /8255
	PC0 /8255	接	继电器
	CS /8255	接	Y1 /IO地址

### 三、编程提示

1、将8254计数器0设置为方式3、计数器1设置为方式0并联使用，CLK0接1MHz时钟，设置两个计数器的初值(乘积为5000000)启动计数器工作后，经过5秒钟OUT1输出高电平。通过8255A口查询OUT1的输出电平，用C口PC0输出开关量控制继电器动作。

2、继电器开关量输入端输入“1”时，继电器常开触点闭合，发光二极管接通，指示灯亮，输入“0”时断开，指示灯灭。

2、参考流程图(见图4-9-2)：

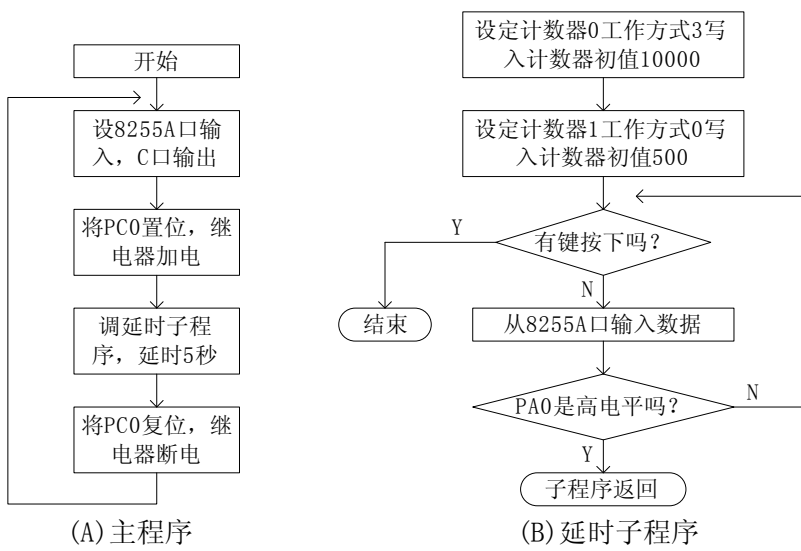


图4-9-2

3、参考程序：JDQ.ASM

```

io8255a equ 280h
io8255b equ 281h
io8255c equ 283h
io8255d equ 288h
io8255e equ 28bh
code segment
assume cs:code
start:
    mov dx, io8255e    ; 设8255为A口输入, C口输出
    mov al, 90h
111:  out dx, al
    mov al, 01         ; 将PC0置位
    out dx, al
    call delay         ; 延时5s
    mov al, 0          ; 将PC0复位
    out dx, al
    call delay         ; 延时5s
    jmp 111            ; 转111
delay proc near        ; 延时子程序

```



```

    push dx
    mov dx, io8255c      ;设8254计数器为方式3
    mov al, 36h
    out dx, al
    mov dx, io8255a
    mov ax, 10000        ;写入计数器初值10000
    out dx, al
    mov al, ah
    out dx, al
    mov dx, io8255c
    mov al, 70h          ;设计数器1为工作方式0
    out dx, al
    mov dx, io8255b
    mov ax, 500           ;写入计数器初值500
    out dx, al
    mov al, ah
    out dx, al
112:  mov ah, 06           ;是否有键按下
    mov dl, 0ffh
    int 21h
    jne exit             ;若有则转exit
    mov dx, io8255d
    in  al, dx            ;查询8255的PA0是否为高电平
    and al, 01
    jz  112               ;若不是则继续
    pop dx
    ret                   ;定时时间到，子程序返回
exit:  mov ah, 4ch
    int 21h
delay endp
code ends
end start

```

## 实验十 存储器读写实验

### 一、实验目的

- 1、熟悉6264静态RAM的使用方法，掌握PC机外存扩充的手段。
- 2、通过对硬件电路的分析，学习了解总线的工作时序。

### 二、实验内容

- 1、硬件电路如下（仅供参考，图中RAM为2K的6264）：

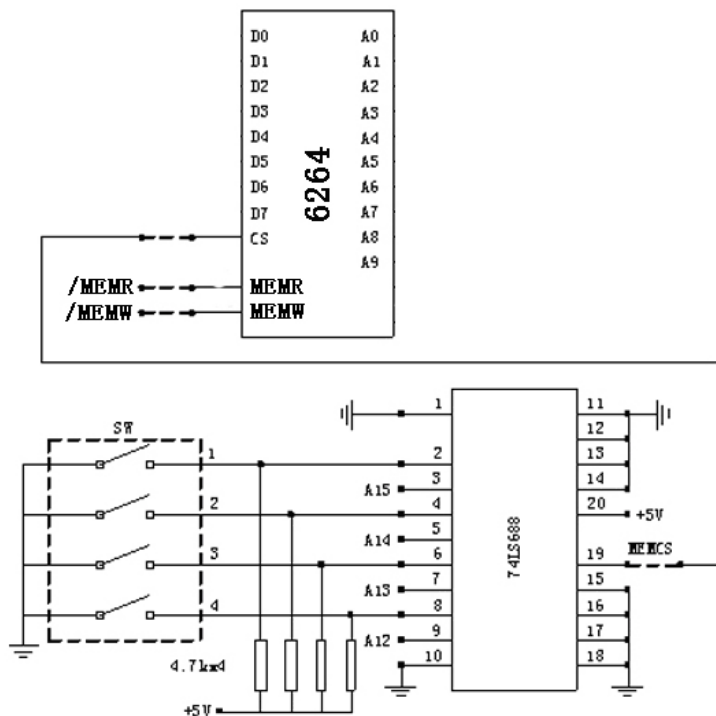


图4-10-1

- 2、编制程序，将字符A-Z循环写入扩展的RAM中，然后再将扩展的RAM内容读出来显示在主机屏幕上。

- 3、接线：
 

/MEMW /总线区	接	MEMW /RAM存储器
/MEMR /总线区	接	MEMR /RAM存储器
/MEMCS/IO译码	接	CS / RAM存储器

### 三、编程提示

- 1、USB接口模块外扩储器的地址范围为0D4000H-0D7fffH。
- 2、通过片选信号的产生方式，确定扩展的RAM在PC机系统中的地址范围。因为段地址已指定，所以其地址为CS=A15 and A14 and A13 and A12，实验台上设有地址选择微动开关，拨动开关，可以选择4000-7fff的地址范围。编制程序，从0d6000H开始循环写入100h个A-Z。开关状态如下：

1	2	3	4	地址
OFF	ON	OFF	OFF	d4000h
OFF	ON	ON	OFF	d6000h

#### 四、程序框图

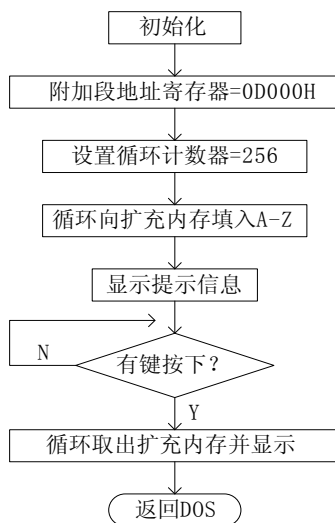


图21-2

#### 五、参考程序

##### 1、MEMRW.ASM

```

data segment
message db 'please enter a key to show the contents!',0dh,0ah,'$'
data ends
code segment
assume cs:code,ds:data,es:data
start:
    mov ax,data
    mov ds,ax
    mov ax,0d000h
    mov es,ax
    mov bx,06000h
    mov cx,100h
    mov dx,40h
repl:  inc dl
      mov es:[bx],dl
      inc bx
      cmp dl,5ah
      jnz ssl
      mov dl,40h
ssl:   loop repl
  
```

```

        mov dx,offset message
        mov ah,09
        int 21h
        mov ah,01h
        int 21h
        mov ax,0d000h
        mov es,ax
        mov bx,06000h
        mov cx,0100h
rep2:   mov dl,es:[bx]
        mov ah,02h
        int 21h
        inc bx
        loop rep2
        mov ax,4c00h
        int 21h
code ends
end start

```

## 实验十一 DMA传送

## 一、实验目的

- 1、掌握PC机工作环境下进行DMA方式数据传送(Block MODE和Demand Mode)(块传送、外部请求传送)方法。
- 2、掌握DMA的编程方法。

## 二、实验原理和内容

- 1、按照图4-11-1将实验箱通用插座D区6264连接好。编程将实验箱的6266缓冲区D4000H, 偏移量为0的一块数据循环写入字符A~Z, 用Block MODE DMA方式传送到实验箱的6264的缓冲区D4A00H上, 并察看送出的数据是否正确。

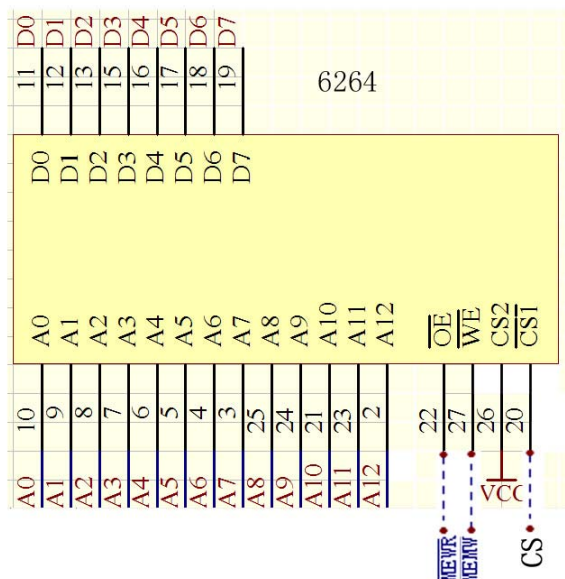


图4-11-1

- 2、用通用插座按图4-11-2连接好电路（74LS74利用实验台上的D触发器）。编程将主机内存缓冲区内D4000H, 偏移量为0的10个数据, 使用Demand Mode DMA方式从内存向外设传送。

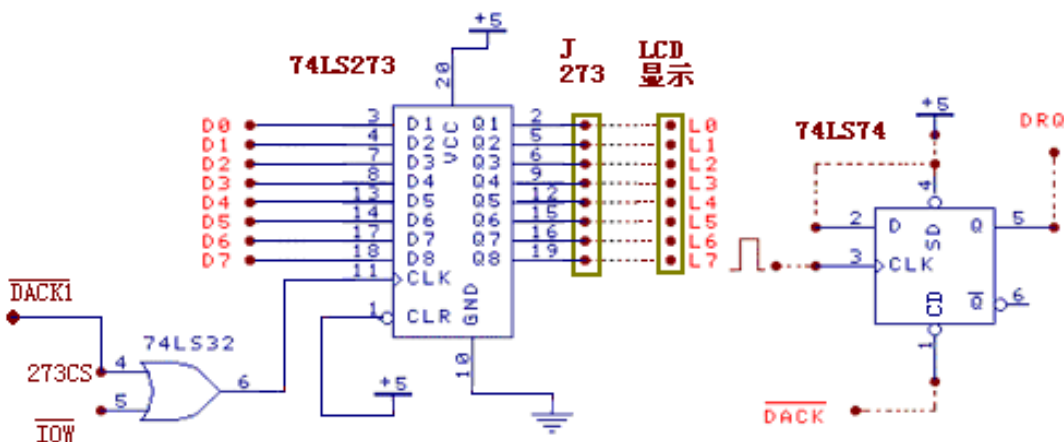


图4-11-2

3、用通用插座按图4-11-3连接好电路（74LS74利用实验台上的D触发器）。编程在主机内存缓冲区D4000H, 偏移量为0的位置开辟数据缓冲区，使用Demand Mode DMA方式从外设向内存传送8个数据并存入数据缓冲区，编程不断显示缓冲区的数据。

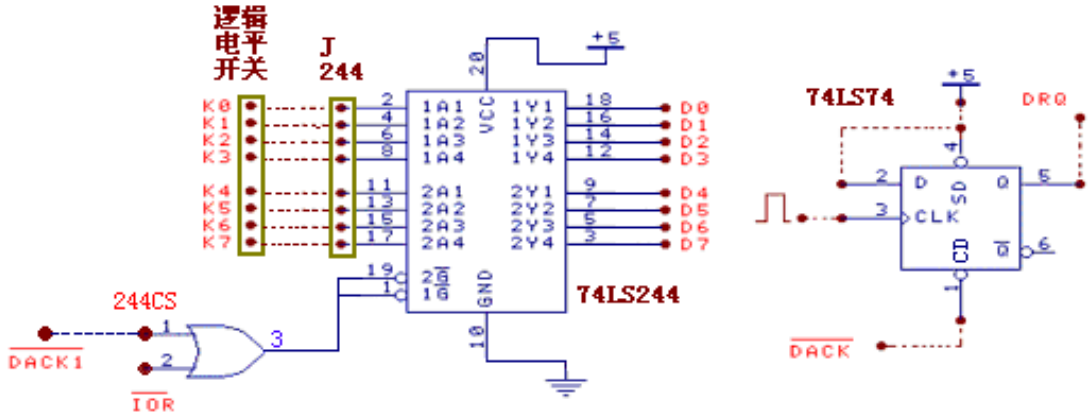



图4-11-3

4、接线：按各实验图接线，图中虚线为实验所需接线

图中  为实验台中单脉冲。

74LS74为实验台上D触发器

### 三、实验提示

1、DMA 请求是由单脉冲输入到 D 触发器，由触发器的 Q 端向DRQ1发出的。CPU响应后发出DACK1，将触发器Q置成低电平以撤消请求。

2、汇编程序中，为避免与系统8237有冲突，TPC-USB模块上的8237端口范围为10H-1F，即按通常模式进行DMA编程时，对8237所有端口均加10H。

### 四、参考流程图（见图4-11-4、图4-11-5、图4-11-6）

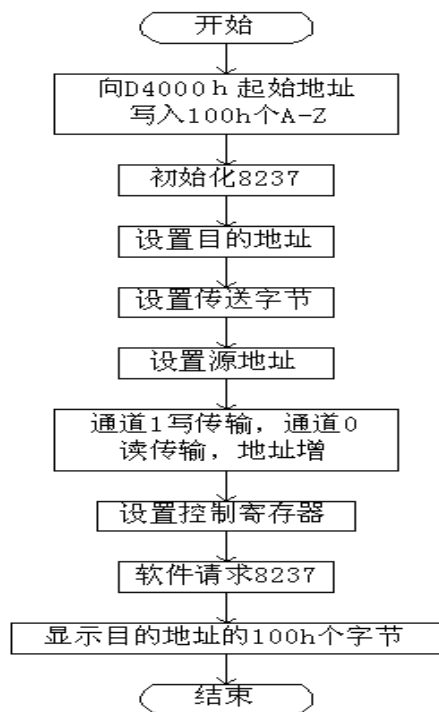


图4-11-4

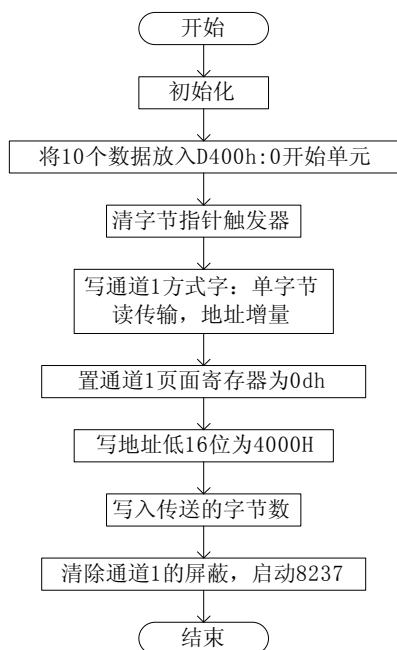


图4-11-5

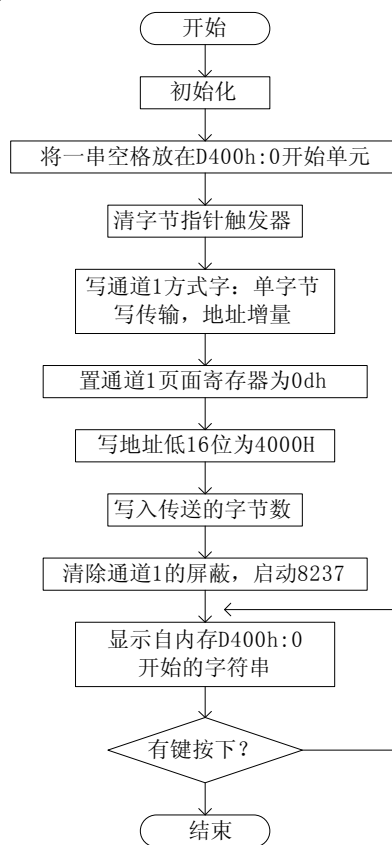


图4-11-6

## 五、参考程序1

### 1、DMA.asm

```
code segment
    assume cs:code
start:
    mov ax, 0D000h
    mov es, ax
    mov bx, 4000h
    mov cx, 0ffh; 100h
    mov dl, 40h
repl:  inc dl
    mov es:[bx], dl
    inc bx
    cmp dl, 5ah
    jnz ss1
    mov dl, 40h
    ss1: loop repl

    mov dx, 18h          ;关闭8237
    mov al, 04h
    out dx, al
    mov dx, 1dh          ;复位
    mov al, 00h
    out dx, al
    mov dx, 12h          ;写目的地址低位
    mov al, 00h
    out dx, al
    mov dx, 12h          ;写目的地址高位
    mov al, 60h;
    out dx, al
    mov dx, 13h          ;传送字节数低位
    mov al, 0ffh;
    out dx, al
    mov dx, 13h          ;传送字节数高位
    mov al, 0; 1h
    out dx, al
    mov dx, 10h          ;源地址低位
    mov al, 00h
    out dx, al
```



```

        mov dx, 10h           ;源地址高位
        mov al, 40h
        out dx, al
        mov dx, 1bh           ;通道1写传输, 地址增
        mov al, 85h
        out dx, al
        mov dx, 1bh           ;通道0读传输, 地址增
        mov al, 88h
        out dx, al
        mov dx, 18h           ;DREQ低电平有效, 存储器到存储器, 开启8237
        mov al, 41h
        out dx, al
        mov dx, 19h           ;通道1请求
        mov al, 04h
        out dx, al
        mov cx, 0F000h
        delay: loop delay
        mov ax, 0D000h;---
        mov es, ax
        mov bx, 06000h
        mov cx, 0ffh;
rep2:    mov dl, es:[bx]
        mov ah, 02h
        int 21h
        inc bx
        loop rep2
        mov ax, 4c00h
        int 21h
        code ends
        end start

```

## 2、参考程序2

DMA\_OUT.asm

```

data segment
outdata db 01, 02, 04, 08, 10h, 20h, 40h, 80h, 0ffh, 00h
data ends
extra segment at 0d400h
ext db 10 dup(?)
extra ends
code segment

```

```

        assume cs:code,ds:data,es:extra
start:
        mov ax,data
        mov ds,ax
        mov ax,extra
        mov es,ax
        lea si,outdata
        lea di,ext
        cld
        mov cx,10
        rep movsb
        out 1ch,al           ;清字节指针
        mov al,49h           ;写方式字
        out 1bh,al
        mov al,0dh           ;置地址页面寄存器
        out 83h,al
        mov al,0             ;写入基地址低十六位
        out 12h,al
        mov al,40h
        out 12h,al
        mov al,0ah           ;写入传送的字节数10
        out 13h,al           ;先写低字节
        mov al,00h
        out 13h,al           ;后写高字节
        mov al,01            ;清通道屏蔽, 启动DMA
        out 1ah,al
        mov ah,4ch
        int 21h
        code ends
        end start

```

### 3、参考程序3

```

DMA_IN.asm
data segment
indata1 db 8 dup(30h),0dh,0ah,24h
data ends
extra segment at 0d400h
indata2 db 11 dup(?)
extra ends
code segment

```

```

        assume cs:code,ds:data,es:extra
start:
        mov ax,data
        mov ds,ax
        mov ax,extra
        mov es,ax
        lea si,indata1
        lea di,indata2
        cld
        mov cx,11
        rep movsb
        mov ax,extra
        mov ds,ax
        mov al,00
        out 1ch,al           ;清字节指针
        mov al,45h          ;写方式字
        out 1bh,al
        mov al,0dh          ;置地址页面寄存器
        out 83h,al
        mov al,00
        out 12h,al          ;写入基地址的低十六位
        mov al,40h
        out 12h,al
        mov ax,7            ;写入传送的8个字节数
        out 13h,al          ;先写低字节
        mov al,ah
        out 13h,al          ;后写高字节
        mov al,01           ;清通道屏蔽
        out 1ah,al          ;启动DMA
sss:    lea dx,indata2
111:    mov ah,09
        int 21h
        mov ah,1
        int 16h
        je sss
exit:   mov ah,4ch
        int 21h
        code ends
        end start

```

## 实验十二 扩展DMA控制器8237

### 一、实验目的

- 1、掌握PC机工作环境下进行DMA方式数据传送(Block MODE和Demand Mode)块传送方法。
- 2、掌握扩展DMA的编程方法。

### 二、实验原理和内容

- 1、图4-12-1将线连接好。编程将实验箱的RAM存储器缓冲区D4000H, 偏移量为0的一块数据循环写入字符A~Z, 用Block MODE DMA方式传送到实验箱的RAM存储器的缓冲区D4A00H上, 并察看送出的数据是否正确。

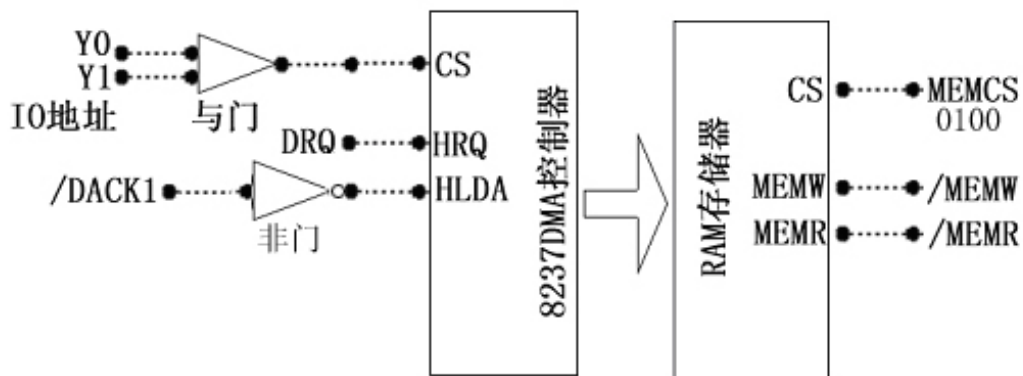
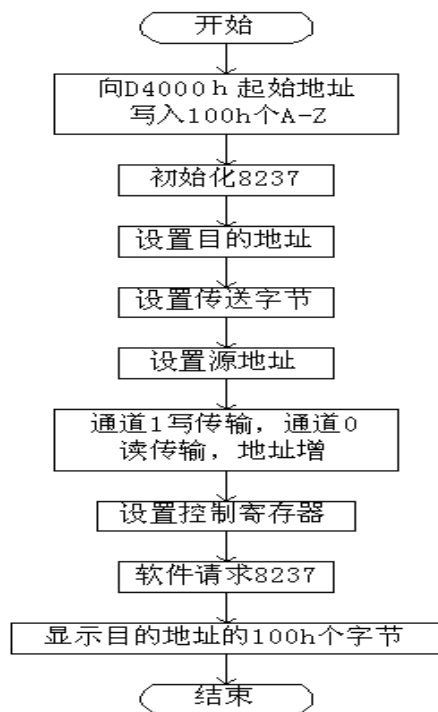


图4-12-1

### 三、实验提示

- 1、DMA 请求是由单脉冲输入到 D 触发器, 由触发器的 Q 端向 DRQ1 发出的。CPU 响应后发出 DACK1, 将触发器 Q 置成低电平以撤消请求。
- 2、汇编程序中, 为避免与系统 8237 有冲突, TPC-USB 模块上的 8237 端口范围为 10H-1F, 即按通常模式进行 DMA 编程时, 对 8237 所有端口均加 10H。
- 3、8237 控制字地址为 16 位, 实验台 IO 地址译每组地址为 8 位, 需要二组连续 IO 地址译码信号。因此实验 8237 的片选信号采用 Y0 和 Y1 经过与门给出。
- 4、核心板的 8237 为主控制器进行级连时, 因 DRQ0 与 USB 接口控制器使用, 因此不要对 8237 控制器进行复位, 对共用的控制寄存器进行初始化。

### 四、流程图



## 五、参考程序

```

;*****;
;*   扩展DMA传送实验（块传送） *;
;*****;

```

```
io8237      equ 280h      ;从8237地址
```

```
code segment
```

```
    assume cs:code
```

```
start:
```

```
    mov ax, 0D000h
```

```
    mov es, ax
```

```
    mov bx, 4000h
```

```
    mov cx, 0ffh; 传输个数
```

```
    mov dl, 40h; 字符A
```

```
rep1:    inc dl
```

```
    mov es:[bx], dl
```

```
    inc bx
```

```
    cmp dl, 5ah
```

```
    jnz ss1
```

```
    mov dl, 40h
```

```
ss1: loop rep1
```

;主8237为主控制器, DRQ0为USB接口控制使用, 不要对  
;其复位和控制寄存器初始化

mov dx, 1bh	;主8237为级联方式
mov al, 0cdh	
out dx, al	
mov al, 01	;清通道屏蔽DRQ1
out 1ah, al	;启动DMA
mov dx, io8237+08h	;关闭从8237
mov al, 04h	
out dx, al	
mov dx, io8237+0dh	;复位从8237
mov al, 00h	
out dx, al	
mov dx, io8237+0ch	;清字节指针
mov al, 00	
out dx, al	
mov dx, io8237+02h	;写目的地址低位
mov al, 00h	
out dx, al	
mov dx, io8237+02h	;写目的地址高位
mov al, 42h	
out dx, al	
mov dx, io8237+03h	;传送字节数低位
mov al, 0ffh	
out dx, al	
mov dx, io8237+03h	;传送字节数高位
mov al, 00h	
out dx, al	
mov dx, io8237+00h	;源地址低位
mov al, 00h	
out dx, al	
mov dx, io8237+00h	;源地址高位
mov al, 40h	
out dx, al	
mov dx, io8237+0bh	;通道1写传输, 地址增
mov al, 85h	
out dx, al	
mov dx, io8237+0bh	;通道0读传输, 地址增
mov al, 88h	
out dx, al	

```

    mov dx, io8237+08h           ;DREQ低电平有效, 存储器到存储器, 开启从8237
    mov al, 41h
    out dx, al
    mov dx, io8237+09h           ;通道1请求
    mov al, 04h
    out dx, al

    mov cx, 0F000h
delay:  loop delay
    mov ax, 0D000h
    mov es, ax
    mov bx, 04200h;目的地址起始
    mov cx, 0ffh;读出字符个数
rep2:mov dl, es:[bx]
    mov ah, 02h
    int 21h
    inc bx
    loop rep2
    mov ax, 4c00h
    int 21h
code ends
end start

```

## 实验十三 中断

### 一、实验目的

- 1、掌握PC机中断处理系统的基本原理。
- 2、学会编写中断服务程序。

### 二、实验原理和内容

#### 1、实验原理

PC机用户可使用的硬件中断只有可屏蔽中断，由8259中断控制器管理。中断控制器用于接收外部的中断请求信号，经过优先级判别等处理后向CPU发出可屏蔽中断请求。IBMPC、PC/XT机内有一片8259中断控制器对外可以提供8个中断源：

中断源	中断类型号	中断功能
IRQ0	08H	时钟
IRQ1	09H	键盘
IRQ2	0AH	保留
IRQ3	0BH	串行口2
IRQ4	0CH	串行口1
IRQ5	0DH	硬盘
IRQ6	0EH	软盘
IRQ7	0FH	并行打印机

8个中断源的中断请求信号线IRQ0~IRQ7在主机62线ISA总线插座中可以引出，系统已设定中断请求信号为“边沿触发”，普通结束方式。对于PC/AT及286以上微机内又扩展了一片8259中断控制，IRQ2用于两片8259之间级连，对外可以提供16个中断源：

中断源	中断类型号	中断功能
IRQ8	070H	实时时钟
IRQ9	071H	用户中断
IRQ10	072H	保留
IRQ11	073H	保留
IRQ12	074H	保留
IRQ13	075H	协处理器
IRQ14	076H	硬盘
IRQ15	077H	保留

TPC-ZK-USB实验系统总线区的IRQ接到了3号中断IRQ3上，即进行中断实验时，所用中断类型为0BH。USB核心板上的IR10接到了10号中断IRQ10上，所用中断类型为072H。

#### 2、实验提示：

- 1)、中断10的优先级要高于3，因为中断10是扩展8259上的中断，扩展8259使用主8259的中断2向主8259申请中断，中断2优先级高于中断3，所以中断10的优先级要高于3
- 2)、由上所述，中断0-15的优先级顺序从高到低为：中断0、中断1、中断8、中断9、中断10、中断11、中断12、中断13、中断14、中断15、中断3、中断4、中断5、中断6、中断7
- 3)、上述中断优先级顺序中，不包含中断2，因为中断2已被扩展8259使用从而扩展



了中断8-15，所以不再出现中断2。

3、接线：      单脉冲2      接      IRQ /总线  
                  单脉冲1      接      IR10/USB核心板上

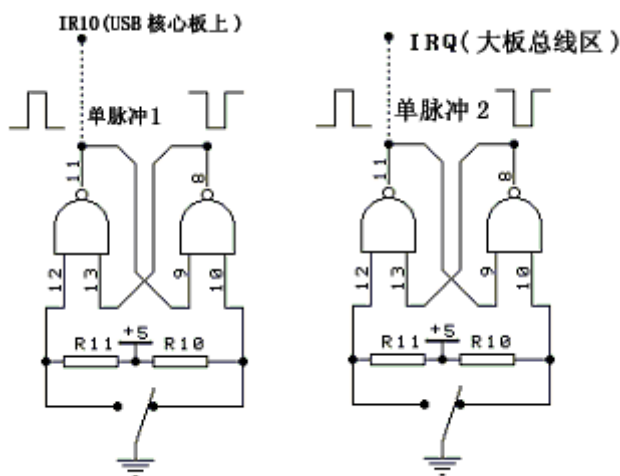


图4-12-1

#### 4、实验内容

1)、中断IRQ3实验，实验电路如图4-12-1，直接用手动产生单脉冲2作为中断请求信号（只需连接一根导线）。要求每按一次开关产生一次中断，在屏幕上显示一次“TPCA Interrupt!”，中断10次后程序退出。

2)、中断IRQ10实验，实验电路如图4-12-1，用手动产生单脉冲1作为中断请求信号，每按一次开关产生一次中断，在屏幕上显示一次“ ”，中断10次后退出。

3)、中断嵌套实验，实验电路如图4-12-1，分别用手动产生单脉冲作为中断IRQ3和IRQ10的请求信号，申请中断IRQ3后，进入中断3程序，再申请高级中级IRQ10。

#### 三、参考流程图

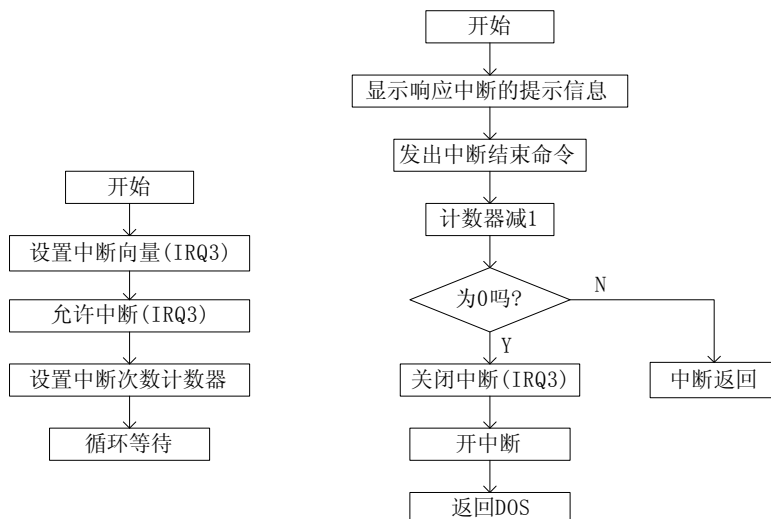


图4-12-2

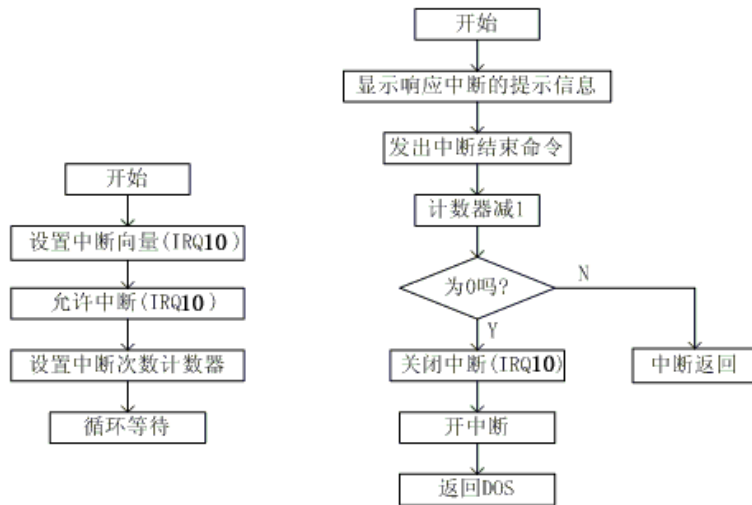


图4-12-3

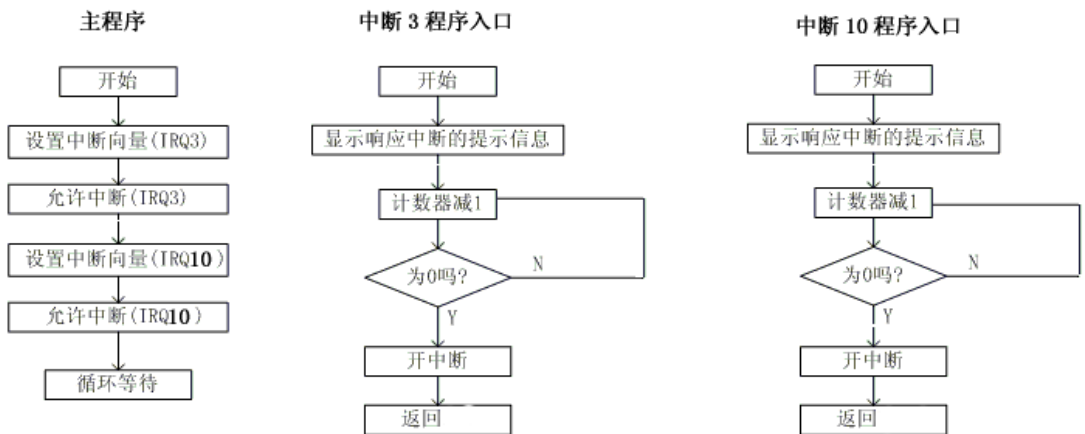


图4-12-4

#### 四、参考程序:

##### 1、INT-IRQ3.ASM

```

;*****
;*          中断IRQ3实验          *
;*****

```

data segment

```

mess db 'TPCA interrupt! ', 0dh, 0ah, '$'
data ends
code segment
assume cs:code, ds:data
start:
    mov ax, cs
    mov ds, ax
    mov dx, offset int3
    mov ax, 250bh
    int 21h                ;设置IRQ3的中断矢量
    in al, 21h              ;读中断屏蔽寄存器
    and al, 0f7h            ;开放IRQ3中断
    out 21h, al
    mov cx, 10              ;记中断循环次数为10次
    sti
11:    jmp 11
int3:                          ;中断服务程序
    mov ax, data
    mov ds, ax
    mov dx, offset mess
    mov ah, 09              ;显示每次中断的提示信息
    int 21h
    mov al, 20h
    out 20h, al              ;发出EOI结束中断
    loop next
    in al, 21h
    or al, 08h               ;关闭IRQ3中断
    out 21h, al
    sti                      ;置中断标志位
    mov ah, 4ch              ;返回DOS
    int 21h
next:    iret
code ends
end start

```

## 2、INT-IRQ10. ASM

```

;*****;
;*      中断IRQ10实验      *;
;*****;

```

```

data segment
mess db 'TPCA interrupt10!',0dh,0ah,'$'
data ends
code segment
assume cs:code,ds:data
start:
    mov ax,cs
    mov ds,ax
    mov dx,offset int10
    mov ax,2572h
    int 21h

    in al,21h
    and al,0fBh
    out 21h,al
    IN AL,0A1H
    AND AL,0FBH
    OUT 0A1H,AL
    mov cx,10
    sti
11:    jmp 11

int10:
    mov ax,data
    mov ds,ax
    mov dx,offset mess
    mov ah,09
    int 21h
    mov al,20h
    out 20h,al
    OUT 0A0H,AL
    loop next
    in al,21h
    or al,08h
    out 21h,al
    sti
    mov ah,4ch
    int 21h
next:    iret

```

```
code ends
```

```
end start
```

### 3、INT-IRQ10+IRQ3.ASM

```
;*****;
```

```
;*      中断嵌套实验      *;
```

```
;*****;
```

;注意:

;1: 中断10的优先级要高于3, 因为中断10是扩展8259上的中断, 扩展8259使用主8259的中断2向主8259申请中断, 中断2优先级高于中断3, 所以中断10的优先级要高于3

;2: 由上所述, 中断0-15的优先级顺序从高到低为: 中断0、中断1、中断8、中断9、中断10、中断11、中断12、中断13、中断14、中断15、中断3、中断4、中断5、中断6、中断7

;3: 上述中断优先级顺序中, 不包含中断2, 因为中断2已被扩展8259使用从而扩展了中断8-15, 所以不再出现中断2

```
data segment
```

```
mess10 db 'TPCA interrupt10 '
```

```
mess10_1 db 30h,' '
```

```
mess10_2 db 30h,' !',0dh,0ah,'$'
```

```
mess3 db 'TPCA interrupt3 '
```

```
mess3_1 db 30h,' '
```

```
mess3_2 db 30h,' !',0dh,0ah,'$'
```

```
data ends
```

```
code segment
```

```
assume cs:code,ds:data
```

```
start:
```

```
.386
```

```
cli
```

```
mov ax,cs
```

```
mov ds,ax
```

```
mov dx,offset int10
```

```
mov ax,2572h
```

```
int 21h
```

```
mov dx,offset int3
```

```
mov ax,250bh
```

```
int 21h
```

```
in al,21h
```

```

    and al, 0f3h
    out 21h, al

    IN AL, 0A1H
    AND AL, 0FbH
    OUT 0A1H, AL
    mov cx, 10
    sti
11:    jmp 11
int10:
    cli
    pushad
    pushfd

    mov ax, data
    mov ds, ax
    mov cx, 10
next10_1:
    mov dx, offset mess10
    mov ah, 09
    int 21h

    add ds:mess10_1, 1
    add ds:mess10_2, 1
    call delay1
    loop next10_1
    mov ds:mess10_2, 30h

    mov al, 20h
    OUT 0A0H, AL
    out 20h, al
    popfd
    popad
    sti
    iret

int3:
    cli
    pushad

```

```

    pushfd
    mov ax, data
    mov ds, ax
    mov cx, 10
next3_1:
    mov dx, offset mess3
    mov ah, 09
    int 21h
    add ds:mess3_1, 1
    add ds:mess3_2, 1
    call delay1
    loop next3_1
    mov ds:mess3_2, 30h
    mov al, 20h
    out 20h, al
    OUT 0A0H, AL
    popfd
    popad
    sti
    iret
delay1    proc
pushad
pushfd
mov cx, 0fh
delay_loop1:
mov bx, 0ffffh
delay_loop2:
dec bx
nop
jnz delay_loop2
loop delay_loop1
popfd
popad
ret
delay1    endp

code ends
end start

```

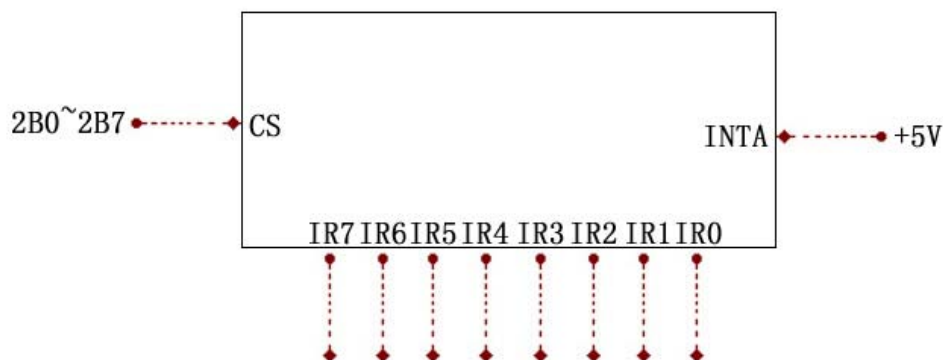
## 实验十四 扩展中断控制器8259

### 一、实验目的

- 1、掌握中断控制器8259管理
- 2、掌握扩展中断

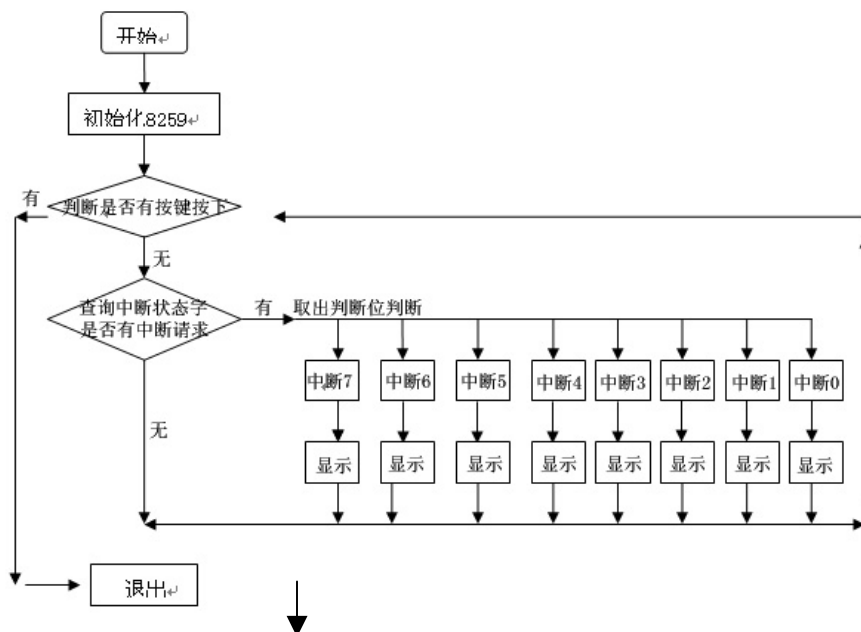
### 二、实验原理和内容

- 1、采用查询方式
- 2、如图4-14-1接线，按单脉冲请求一次中断，屏幕上显示相应的中断请求号



- 3、接线：
- |          |   |                     |
|----------|---|---------------------|
| Y6 /IO地址 | 接 | CS /8259            |
| +5V      | 接 | INTA /8259          |
| 单脉冲      | 接 | IR0/IR1/……IR7 /8259 |

### 三、参考流程图





#### 四、参考程序

```
;8259-1.asm
;*****
;      8259中断查询方式应用实验
;*****
I8259_1 EQU 2B0H      ;8259的ICW1端口地址
I8259_2 EQU 2B1H      ;8259的ICW2端口地址
I8259_3 EQU 2B1H      ;8259的ICW3端口地址
I8259_4 EQU 2B1H      ;8259的ICW4端口地址
O8259_1 EQU 2B1H      ;8259的OCW1端口地址
O8259_2 EQU 2B0H      ;8259的OCW2端口地址
O8259_3 EQU 2B0H      ;8259的OCW3端口地址

data segment

    mes1 db 'you can play a key on the keyboard!',0dh,0ah,24h
    mes2 dd mes1
    mess1 db 'Hello! This is interrupt * 0 *!',0dh,0ah,'$'
    mess2 db 'Hello! This is interrupt * 1 *!',0dh,0ah,'$'
    mess3 db 'Hello! This is interrupt * 2 *!',0dh,0ah,'$'
    mess4 db 'Hello! This is interrupt * 3 *!',0dh,0ah,'$'
    mess5 db 'Hello! This is interrupt * 4 *!',0dh,0ah,'$'
    mess6 db 'Hello! This is interrupt * 5 *!',0dh,0ah,'$'
    mess7 db 'Hello! This is interrupt * 6 *!',0dh,0ah,'$'
    mess8 db 'Hello! This is interrupt * 7 *!',0dh,0ah,'$'
data ends

stacks segment
    db 100 dup(?)
stacks ends
STACK1 SEGMENT STACK
    DW 256 DUP(?)
STACK1 ENDS

CODE SEGMENT
    ASSUME CS:CODE, DS:DATA, SS:stacks, ES:DATA
START: mov ax, data
        mov ds, ax
        mov es, ax
```

```

mov ax, stacks
mov ss, ax
MOV DX, I8259_1      ;初始化8259的ICW1
MOV AL, 13H          ;边沿触发、单片8259、需要ICW4
OUT DX, AL

MOV DX, I8259_2      ;初始化8259的ICW4
MOV AL, 0B0H          ;非自动结束EOI
OUT DX, AL
;MOV AL, 00H
;OUT DX, AL
MOV AL, 03H
OUT DX, AL
MOV DX, 08259_1      ;初始化8259的OCW1
MOV AL, 00H           ;打开IR0和IR1的屏蔽位
OUT DX, AL

QUERY: MOV AH, 1      ;判断是否有按键按下
INT 16H
JNZ QUIT              ;有按键则退出
MOV DX, 08259_3       ;向8259的OCW3发送查询命令
MOV AL, 0CH
OUT DX, AL
IN AL, DX              ;读出查询字
MOV AH, AL
AND AL, 80H
TEST AL, 80H          ;判断中断是否已响应
JZ QUERY              ;没有响应则继续查询
MOV AL, AH
AND AL, 07H
CMP AL, 00H
JE IR0ISR              ;若为IR0请求，跳到IR0处理程序
CMP AL, 01H
JE IR1ISR              ;若为IR1请求，跳到IR1处理程序
CMP AL, 02H
JE IR2ISR
CMP AL, 03H
JE IR3ISR
CMP AL, 04H

```

```

        JE    IR4ISR
        CMP   AL, 05H
        JE    IR5ISR
        CMP   AL, 06H
        JE    IR6ISR
        CMP   AL, 07H
        JE    IR7ISR
        JMP   QUERY
IR0ISR: MOV   AX, DATA
        MOV   DS, AX
        MOV   DX, offset mess1      ;显示提示信息
        MOV   AH, 09
        INT   21H
        JMP   EOI
IR1ISR: MOV   AX, DATA
        MOV   DS, AX
        MOV   DX, offset mess2      ;显示提示信息
        MOV   AH, 09
        INT   21H
        JMP   EOI
IR2ISR: MOV   AX, DATA
        MOV   DS, AX
        MOV   DX, offset mess3      ;显示提示信息
        MOV   AH, 09
        INT   21H
        JMP   EOI
IR3ISR: MOV   AX, DATA
        MOV   DS, AX
        MOV   DX, offset mess4      ;显示提示信息
        MOV   AH, 09
        INT   21H
        JMP   EOI
IR4ISR: MOV   AX, DATA
        MOV   DS, AX
        MOV   DX, offset mess5      ;显示提示信息
        MOV   AH, 09
        INT   21H
        JMP   EOI
IR5ISR: MOV   AX, DATA

```

```

        MOV DS, AX
        MOV DX, offset mess6      ;显示提示信息
        MOV AH, 09
        INT 21H
        JMP EOI
IR6ISR: MOV AX, DATA
        MOV DS, AX
        MOV DX, offset mess7      ;显示提示信息
        MOV AH, 09
        INT 21H
        JMP EOI
IR7ISR: MOV AX, DATA
        MOV DS, AX
        MOV DX, offset mess8      ;显示提示信息
        MOV AH, 09
        INT 21H
EOI:
        MOV DX, 08259_2           ;向8259发送中断结束命令
        MOV AL, 20H
        OUT DX, AL
        JMP QUERY
QUIT:  MOV AX, 4C00H              ;结束程序退出
        INT 21H
CODE ENDS
        END START

```

## 实验十五 可编程并行接口8255方式 1

### 一、实验目的

- 1、掌握8255工作方式 1 时的使用及编程。
- 2、进一步掌握中断处理程序的编写。

### 二、实验原理和内容

- 1、按图4-15-1，8255方式1的输出电路连好线路。
- 2、编程:每按一次单脉冲按钮产生一个正脉冲使8255产生一次中断请求，让CPU进行一次中断服务:依次输出01H、02H，04H，08H，10H，20H，40H，80H使L0~L7依次发光，中断 8 次结束。
- 3、按图4-15-2，8255方式1输入电路，连好线路。

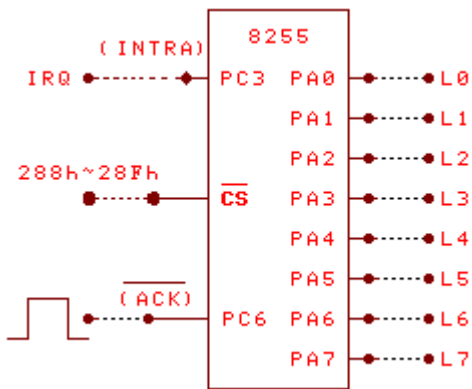


图4-15-1 输出电路

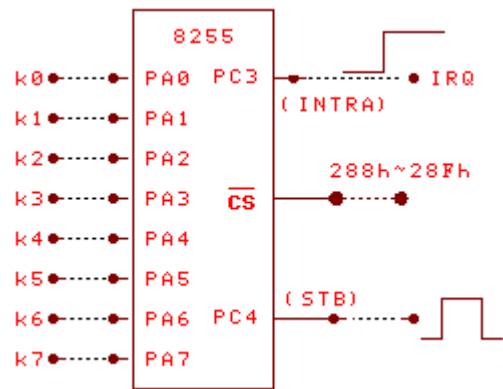


图4-15-2 输入电路

- 4、编程:每按一次单脉冲按钮产生一个正脉冲使8255产生一次中断请求，让CPU进行一次中断服务:读取逻辑电平开关预置的ASCII码，在屏幕上显示其对应的字符，中断 8 次结束。
- 5、接线:按图 (A) 或 (B) 接线，图中虚线为实验所需接线。

### 三、参考流程图: (如图4-15-3、图4-15-4)

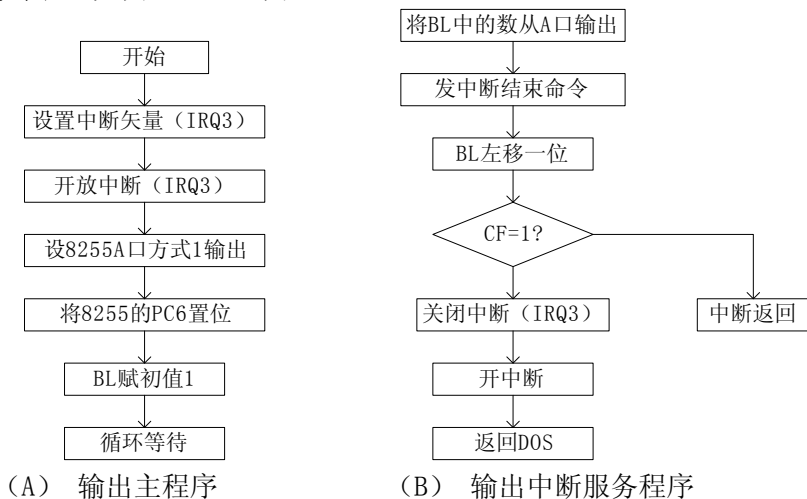


图4-15-3

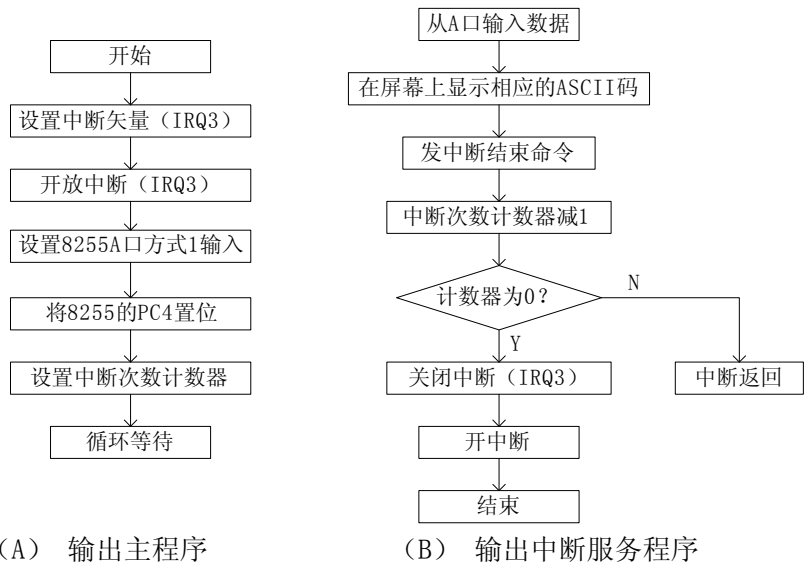


图4-15-4

#### 四、参考程序

##### 1、E8255-10UT. ASM

```

code segment
assume cs:code
start:
    mov ax,cs
    mov ds,ax
    mov dx,offset int_proc
    mov ax,250bh          ;设外部中断int_proc类型为0BH
    int 21h
    mov dx,21h
    in al,dx
    and al,0f7h           ;开放IRQ3中断
    out dx,al
    mov dx,28bh           ;置8255为A口方式1输出
    mov al,0a0h
    out dx,al
    mov al,0dh            ;将PC6置位
    out dx,al
    mov bl,1
11:    jmp 11              ;循环等待
int_proc:

```

```

        mov al,bl
        mov dx,288h          ;将AL从8255的A口输出
        out dx,al
        mov al,20h
        out 20h,al
        shl bl,1
        jnc next              ;中断次数小于8，返回主程序
        in al,21h
        or al,08h              ;关闭IRQ7中断
        out 21h,al
        sti                    ;开中断
        mov ah,4ch             ;返回DOS
        int 21h
next:    iret
code ends
end start

```

## 2、E8255-1IN.ASM

```

code segment
assume cs:code
start:
        mov ax,cs
        mov ds,ax
        mov dx,offset int_proc ;设置IRQ3中断矢量
        mov ax,250bh
        int 21h
        mov dx,21h
        in al,dx
        and al,0f7h            ;开放IRQ3中断
        out dx,al
        mov dx,28bh            ;设8255为A口方式1输入
        mov al,0b8h
        out dx,al
        mov al,09h
        out dx,al
        mov bl,8                ;BL为中断次数计数器
11:     jmp 11

```

int_proc:	;中断服务程序
mov dx, 288h	;自8255A口输入一数据
in al, dx	
mov dl, al	;将所输入的数据保存到DL
mov ah, 02h	;显示ASCII码为DL的字符
int 21h	
mov dl, 0dh	;回车
int 21h	
mov dl, 0ah	;换行
int 21h	
mov dx, 20h	;发出EOI结束命令
mov al, 20h	
out dx, al	
dec bl	;计数器减1
jnz next	;不为0则返回主程序
in al, 21h	
or al, 08h	
out 21h, al	;关IRQ3中断
sti	;开中断
mov ah, 4ch	;返回DOS
int 21h	
next:   iret	
code ends	
end start	



## 实验十六 串行通讯8251

### 一、实验目的

- 1、了解串行通讯的基本原理。
- 2、掌握串行接口芯片8251的工作原理和编程方法。

### 二、实验原理和内容

- 1、按图4-16-1连接好电路, (8251插通用插座) 其中8254计数器用于产生8251的发送和接收时钟, TXD和RXD连在一起。
- 2、编程: 从键盘输入一个字符, 将其ASCII码加 1 后发送出去, 再接收回来在屏幕上显示, 实现自发自收。

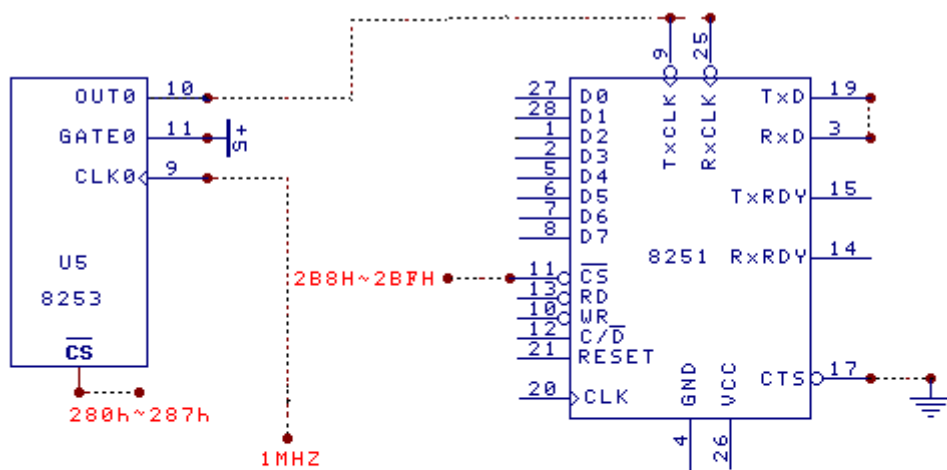


图4-16-1 串行通讯电路

3、接线:	CLK0 /8254	接	1M时钟
	GATE0 /8254	接	+5V
	OUT0 /8254	接	TX/RXCLK /8251
	CS /8254	接	Y0 /IO地址
	CS /8251	接	Y7 /IO地址
	RXD /8251	接	TXD /8251

### 三、实验提示

- 1、图示电路8251的控制口地址为2B9H, 数据口地址为2B8H。

2、8254计数器的计数初值=时钟频率/(波特率×波特率因子)，这里的时钟频率接1MHz，波特率若选1200，波特率因子若选16，则计数器初值为52。

3、收发采用查询方式。

#### 四、参考流程图（见图4-16-2）

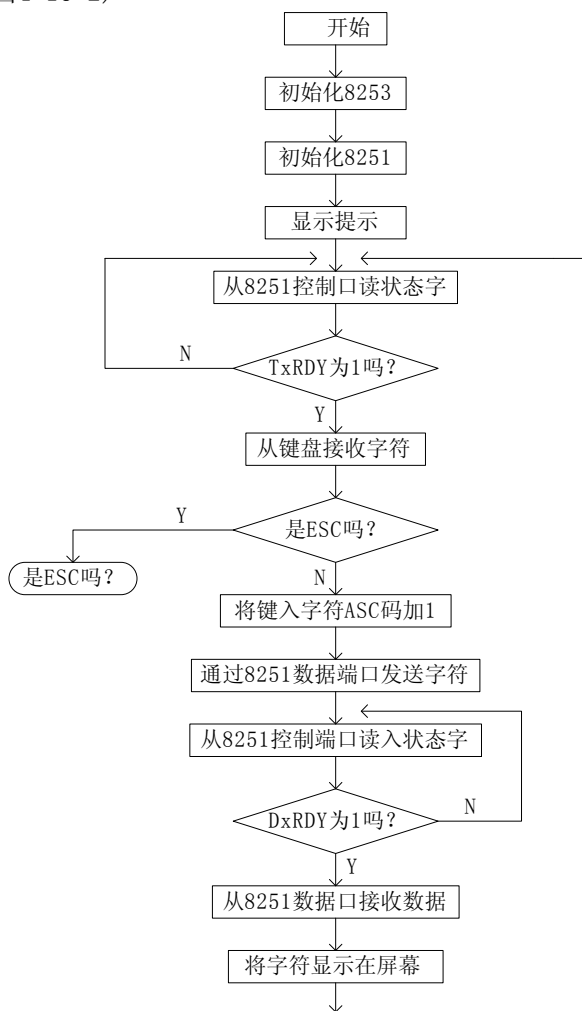


图4-16-2

#### 五、参考程序：

##### 1、 E8251.ASM

```
data segment
io8254a equ 280h
io8254b equ 283h
io8251a equ 2b8h
io8251b equ 2b9h
```

```

mes1    db  'you can play a key on the keyboard!', 0dh, 0ah, 24h
mes2    dd  mes1
data ends
code segment
assume cs:code, ds:data
start:

        mov ax, data
        mov ds, ax
        mov dx, io8254b      ;设置8254计数器0工作方式
        mov al, 16h
        out dx, al
        mov dx, io8254a
        mov al, 52           ;给8254计数器0送初值
        out dx, al
        mov dx, io8251b      ;初始化8251
        xor al, al
        mov cx, 03           ;向8251控制端口送3个0
delay:  call out1
        loop delay
        mov al, 40h          ;向8251控制端口送40H, 使其复位
        call out1
        mov al, 4eh          ;设置为1个停止位, 8个数据位, 波特率因子为16
        call out1
        mov al, 27h          ;向8251送控制字允许其发送和接收
        call out1
        lds dx, mes2         ;显示提示信息
        mov ah, 09
        int 21h
waiti:  mov dx, io8251b
        in al, dx
        test al, 01          ;发送是否准备好
        jz waiti
        mov ah, 01           ;是, 从键盘上读一字符
        int 21h
        cmp al, 27           ;若为ESC, 结束
        jz exit
        mov dx, io8251a
        inc al
        out dx, al           ;发送

```

```

        mov cx, 40h
s51:    loop s51                ;延时
next:   mov dx, io8251b
        in al, dx
        test al, 02            ;检查接收是否准备好
        jz next                ;没有, 等待
        mov dx, io8251a
        in al, dx              ;准备好, 接收
        mov dl, al
        mov ah, 02             ;将接收到的字符显示在屏幕上
        int 21h
        jmp waiti
exit:   mov ah, 4ch             ;退出
        int 21h
out1 proc near                  ;向外发送一字节的子程序
        out dx, al
        push cx
        mov cx, 40h
gg:     loop gg                 ;延时
        pop cx
        ret
out1 endp
code ends
end start

```

## 实验十七 数/模转换器

### 一、实验目的

了解数/模转换器的基本原理，掌握DAC0832芯片的使用方法。

### 二、实验原理和内容

1、实验电路原理如图4-17-1，DAC0832采用单缓冲方式，具有单双极性输出端(图中的U<sub>a</sub>、U<sub>b</sub>)，利用debug输出命令(Out 290 数据)输出数据给DAC0832，用万用表测量单极性输出端U<sub>a</sub>及双极性输出端U<sub>b</sub>的电压，验证数字与电压之间的线性关系。

2、编程产生以下波形(从U<sub>b</sub>输出，用示波器观察)

(1) 锯齿波

(2) 正弦波

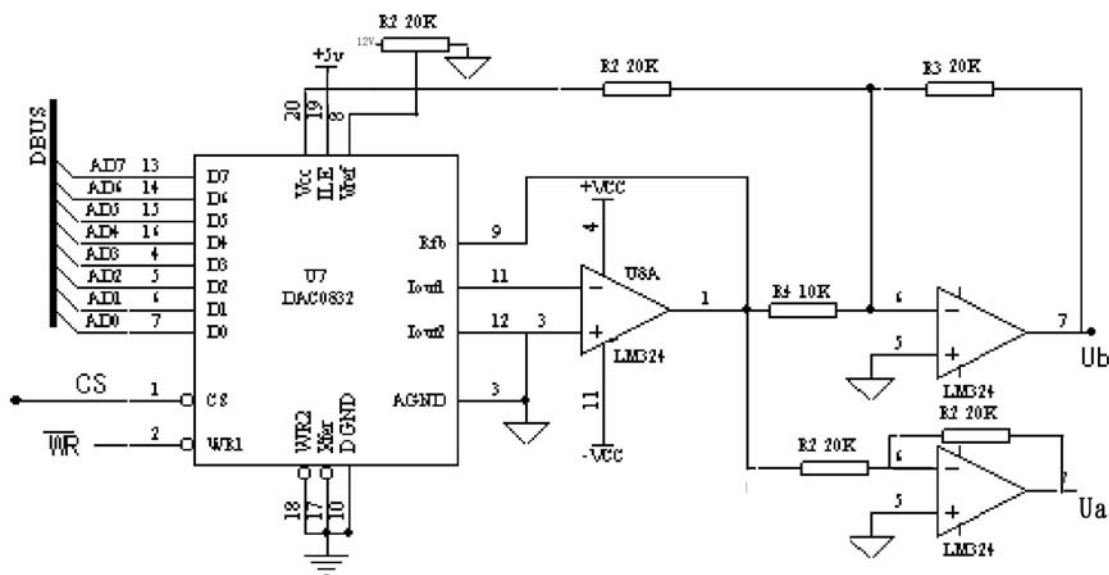


图4-17-1

3、接线： CS /0832 接 Y2 /IO地址

### 三、编程提示

1、8位D/A转换器DAC0832的口地址为290H，输入数据与输出电压的关系为：

$$U_a = \frac{U_{REF}}{256} \times N$$

$$U_b = -\frac{2U_{REF}}{256} \times N - 5$$

(U<sub>REF</sub>表示参考电压,N表示数据)，这里的参考电压为可调电位器调节的电位(为了便于编程出厂为5.12V)。

2、产生锯齿波只须将输出到DAC0832的数据由0循环递增。产生正弦波可根据正弦函数建一个下弦数字量表，取值范围为一个周期，表中数据个数在16个以上。

四、参考流程图：

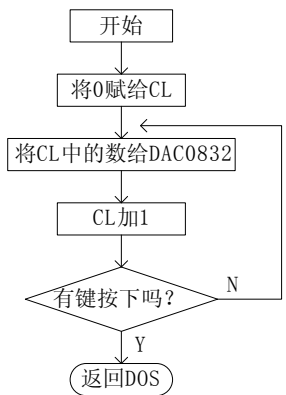


图4-17-2 锯齿波

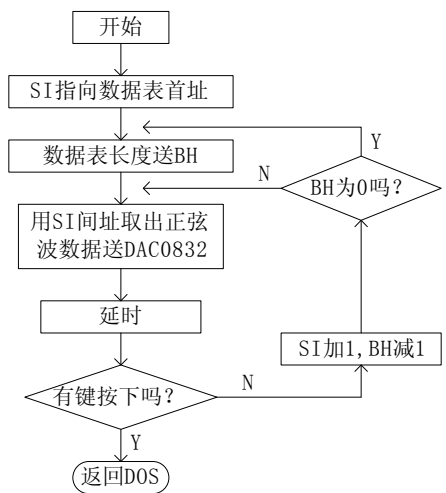


图4-17-3 正弦波

五、参考程序

1、DA\_1.ASM

```
io0832a equ 290h
code segment
assume cs:code
start:
    mov cl,0
    mov dx,io0832a
l11:  mov al,cl
    out dx,al
    inc cl           ;c1加1
    inc cl
    inc cl
    inc cl
    inc cl
    inc cl
    inc cl
    push dx
    mov ah,06h      ;判断是否有键按下
    mov dl,0ffh
    int 21h
    pop dx
    jz l11          ;若无则转LLL
    mov ah,4ch      ;返回
    int 21h
```

```
code ends
end start
```

## 2、参考程序 2

```
DA_2.ASM
data segment
io0832a equ 290h
sin      db 80h, 96h, 0aeh, 0c5h, 0d8h, 0e9h, 0f5h, 0fdh
          db 0ffh, 0fdh, 0f5h, 0e9h, 0d8h, 0c5h, 0aeh, 96h
          db 80h, 66h, 4eh, 38h, 25h, 15h, 09h, 04h
          db 00h, 04h, 09h, 15h, 25h, 38h, 4eh, 66h      ;正弦波数据
data ends
code segment
assume cs:code, ds:data
start:
    mov ax, data
    mov ds, ax
11:    mov si, offset sin      ;置正弦波数据的偏移地址为SI
    mov bh, 32                ;一组输出32个数据
111:   mov al, [si]            ;将数据输出到D/A转换器
    mov dx, io0832a
    out dx, al
    mov ah, 06h
    mov dl, 0ffh
    int 21h
    jne exit
    mov cx, 1
delay: loop delay             ;延时
    inc si                    ;取下一个数据
    dec bh
    jnz 111                   ;若未取完32个数据则转111
    jmp 11
exit:  mov ah, 4ch             ;退出
    int 21h
code ends
end start
```

## 实验十八 模/数转换器0809

### 一、实验目的

了解模/数转换的基本原理，掌握ADC0809的使用方法。

### 二、实验原理和内容

1、实验电路原理图如图4-18-1。通过实验台左下角电位器RW1输出0~5V 直流电压送入ADC0809通道0(IN0), 利用debug的输出命令启动 A/D 转换器，输入命令读取转换结果，验证输入电压与转换后数字的关系。

启动IN0开始转换：0 0298 0

读取转换结果： I 0298

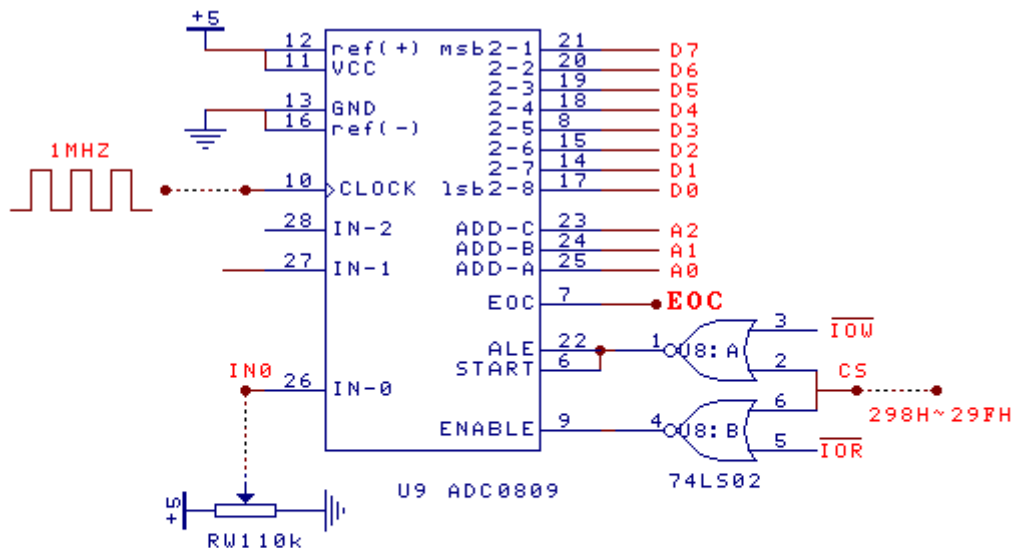


图4-18-1 模数转换电路

2、编程采集IN0输入的电压, 在屏幕上显示出转换后的数据(用16进制数)。

3、给IN1输入一个低频变化信号(幅度为0~5V)，编程采集这个信号数据并在屏幕上显示波形。

4、接线： CS /0809 接 Y3 /IO地址  
IN0 /0809 接 0~5V /直流信号

### 三、实验提示

1、ADC0809的IN0口地址为298H，IN1口地址为299H。

2、IN0单极性输入电压与转换后数字的关系为：



$$N = \frac{U_i}{U_{REF}/256}$$

其中 $U_i$ 为输入电压， $U_{REF}$ 为参考电压，这里的参考电压为可调节电位器调节出的电压，为了便于编程出厂时参考电压为。

3、一次A/D转换的程序可以为

```
MOV     DX, 口地址
OUT     DX, AL      ; 启动转换
; 延时
IN      AL, DX      ; 读取转换结果放在AL中
```

四、参考流程图（见图4-18-2、图4-18-3）

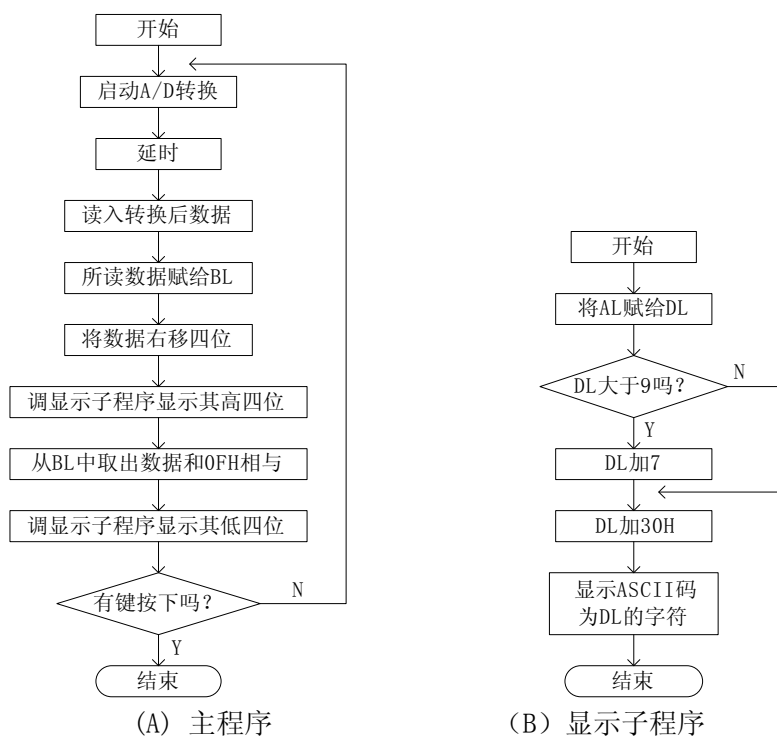


图4-18-2

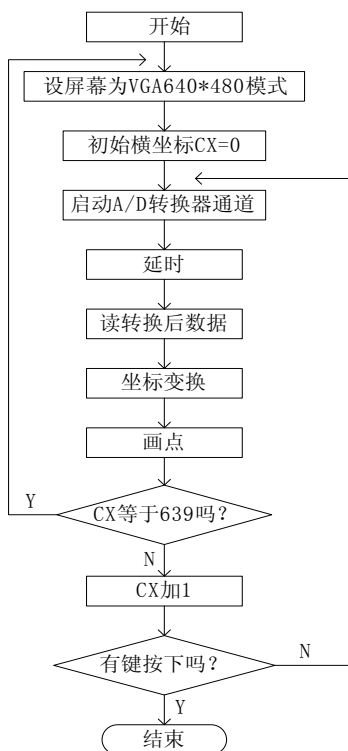


图4-18-3

## 五、参考程序

### 1、参考程序1

AD\_1.ASM

```

io0809a      equ 298h
code segment
assume cs:code
start:
    mov dx,io0809a        ;启动A/D转换器
    out dx,al
    mov cx,0ffh           ;延时
delay: loop delay
    in  al,dx              ;从A/D转换器输入数据
    mov bl,al              ;将AL保存到BL
    mov cl,4
    shr al,cl              ;将AL右移四位
    call disp              ;调显示子程序显示其高四位
    mov al,bl
    and al,0fh
    call disp              ;调显示子程序显示其低四位
    mov ah,02

```

```

        mov dl, 20h                ;加回车符
        int 21h
        mov dl, 20h
        int 21h
        push dx
        mov ah, 06h                ;判断是否有键按下
        mov dl, 0ffh
        int 21h
        pop dx
        je start                    ;若没有转START
        mov ah, 4ch                ;退出
        int 21h
disp proc near                      ;显示子程序
        mov dl, al
        cmp dl, 9                  ;比较DL是否>9
        jle ddd                    ;若不大于则为'0'-'9',加30h为其ASCII码
        add dl, 7                   ;否则为'A'-'F',再加7
ddd:    add dl, 30h                 ;显示
        mov ah, 02
        int 21h
        ret
disp endp
code ends
end start

```

## 2、参考程序 2

AD\_2.ASM

```

io0809b equ 299h
code segment
assume cs:code
start:  mov ax, 0012h              ;设屏幕显示方式为VGA 640X480模式
        int 10h
start1: mov ax, 0600h
        int 10h                    ;清屏
        and cx, 0                  ;cx为横坐标
draw:   mov dx, io0809b            ;启动A/D转换器通道1
        out dx, al
        mov bx, 200                ;延时
delay:  dec bx
        jnz delay

```

```

in al,dx                ;读入数据
mov ah,0
mov dx,368              ;dx为纵坐标
sub dx,ax
mov al,0ah              ;设置颜色
mov ah,0ch              ;画点
int 10h
cmp cx,639              ;一行是否满
jz start1               ;是则转start
inc cx                  ;继续画点
push dx
mov ah,06h              ;是否有键按下
mov dl,0ffh
int 21h
pop dx
je draw                 ;无,则继续画点
mov ax,0003             ;有恢复屏幕为字符方式
int 10h
mov ah,4ch              ;返回
int 21h
code ends
end start

```

# 实验十九 步进电机控制实验

## 一、实验目的

- 1、了解步进电机控制的基本原理。
- 2、掌握控制步进电机转动的编程方法。

## 二、实验原理和内容

- 1、按图4-19-1连接线路，利用8255输出脉冲序列，开关K0~K6控制步进电机转速，K7控制步进电机转向。8255 CS接288H~28FH。PA0~PA3接BA~BD；PC0~PC7接K0~K7。
- 2、编程:当K0~K6中某一开关为“1”（向上拨）时步进电机启动。K7向上拨电机正转，向下拨电机反转。

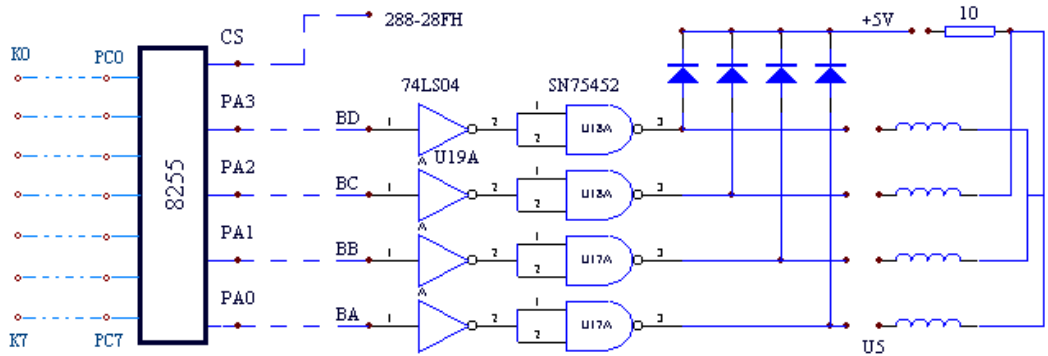


图4-19-1

- 3、接线： PA7~PA0 /8255 接 K7~K0 /逻辑电平开关
- CS /8255 接 Y1 /IO地址
- PC3~PC0 /8255 接 BD~BA /步进电机

## 三、实验说明

步进电机驱动原理是通过对每相线圈中的电流的顺序切换来使电机作步进式旋转。驱动电路由脉冲信号来控制，所以调节脉冲信号的频率便可改变步进电机的转速。

如图4-19-2所示：本实验使用的步进电机用直流+5V电压，每相电流为0.16A，电机线圈由四相组成，即：φ1(BA)；φ2(BB)；φ3(BC)；φ4(BD)

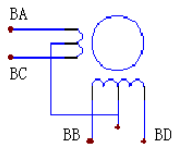


图4-19-2

驱动方式为二相激励方式，各线圈通电顺序如下表。

顺序 \ 相	φ 1	φ 2	φ 3	φ 4
0	1	1	0	0
1	0	1	1	0

反时针方向回转



正时针方向回转

2	0	0	1	1
3	1	0	0	1

表中首先向 $\phi 1$ 线圈— $\phi 2$ 线圈输入驱动电流，接着 $\phi 2-\phi 3$ ， $\phi 3-\phi 4$ ， $\phi 4-\phi 1$ ，又返回到 $\phi 1-\phi 2$ ，按这种顺序切换，电机轴按顺时针方向旋转。

实验可通过不同长度延时来得到不同频率的步进电机输入脉冲，从而得到多种步进速度。

#### 四、参考流程图

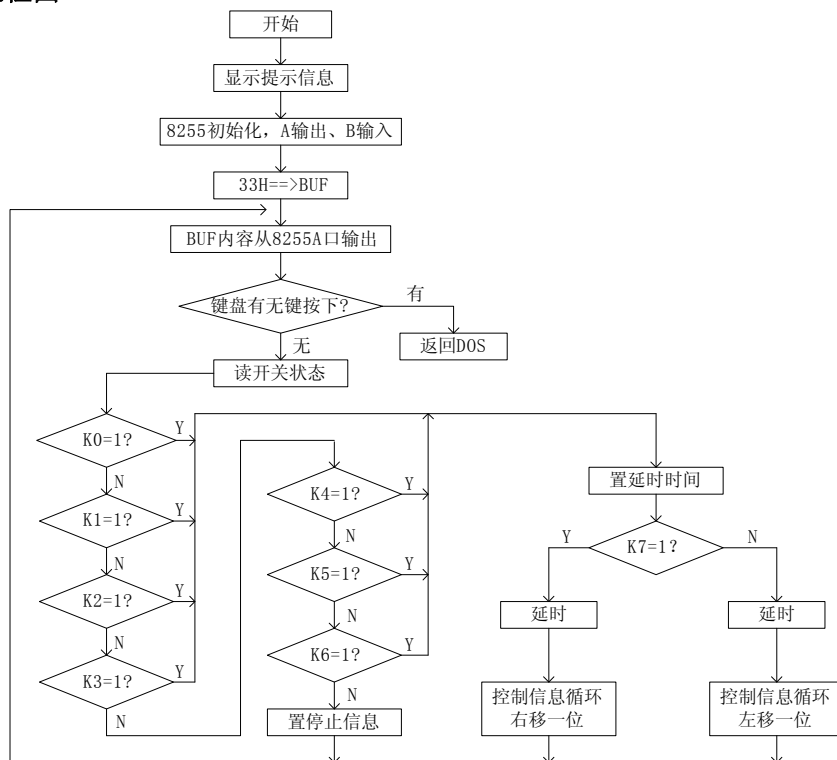


图4-19-3

#### 五、参考程序:

##### 1、BJDJ. ASM

```

data segment
p55a    equ 288h    ;8255 a port output
p55c    equ 28ah    ;8255 c port input
p55ctl  equ 28bh    ;8255 coutrl port
buf     db  0
mes     db  'k0-k6 are speed contyol',0ah,0dh
        db  'k6 is the lowest speed ',0ah,0dh
        db  'k0 is the highest speed',0ah,0dh
        db  'k7 is the direction control',0ah,0dh,'$'

data ends

code segment
assume cs:code,ds:data

```

```

start:
    mov ax,cs
    mov ds,ax
    mov ax,data
    mov ds,ax
    mov dx,offset mes
    mov ah,09
    int 21h
    mov dx,p55ctl
    mov al,8bh
    out dx,al                ;8255 c input, a output
    mov buf,33h
out1:  mov al,buf
    mov dx,p55a
    out dx,al
    push dx
    mov ah,06h
    mov dl,0ffh
    int 21h                ;any key pressed
    pop dx
    je in1
    mov ah,4ch
    int 21h
in1:   mov dx,p55c
    in al,dx                ;input switch value
    test al,01h
    jnz k0
    test al,02h
    jnz k1
    test al,04h
    jnz k2
    test al,08h
    jnz k3
    test al,10h
    jnz k4
    test al,20h
    jnz k5
    test al,40h
    jnz k6

```

```

stop:  mov dx,p55a
        mov al,0ffh
        jmp out1
k0:    mov bl,10h
sam:   test al,80h
        jz zx0
        jmp nx0
k1:    mov bl,18h
        jmp sam
k2:    mov bl,20h
        jmp sam
k3:    mov bl,40h
        jmp sam
k4:    mov bl,80h
        jmp sam
k5:    mov bl,0c0h
        jmp sam
k6:    mov bl,0ffh
        jmp sam
zx0:   call delay
        mov al,buf
        ror al,1
        mov buf,al
        jmp out1
nx0:   call delay
        mov al,buf
        rol al,1
        mov buf,al
        jmp out1
delay proc near
delay1: mov cx,05a4h
delay2: loop delay2
        dec bl
        jnz delay1
        ret
delay endp
code ends
end start

```



## 实验二十 直流电机转速控制实验

### 一、实验目的

- 1、进一步了解DAC0832的性能及编程方法。
- 2、了解直流电机控制的基本方法。

### 二、实验原理和内容

- 1、按图4-20-1线路接线。DAC0832的CS接290H~297H, Ub接DJ插孔, 8255 CS接288H~28FH。
- 2、编程利用DAC0832输出一串脉冲, 经放大后驱动小直流电机, 利用开关K0~K5控制改变输出脉冲的电平及持续时间, 达到使电机加速, 减速之目的。

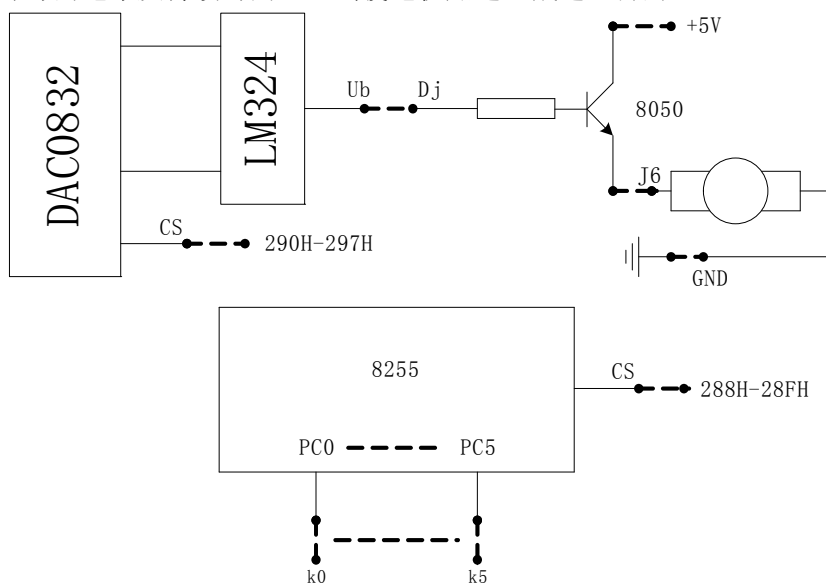


图4-20-1

- 3、接线：
- |               |   |               |
|---------------|---|---------------|
| CS /0832      | 接 | Y2 /IO地址      |
| UB /0832      | 接 | 直流电机          |
| CS /8255      | 接 | Y1 /IO地址      |
| PC7~PC0 /8255 | 接 | K7~K0 /逻辑电平开关 |

### 三、实验原理简述

小直流电机的转速是由Ub输出脉冲的占空比来决定的, 正向占空比越大转速越快, 反之越慢。见下图4-20-2例:

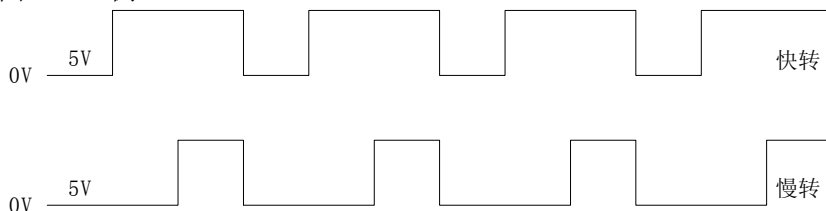


图4-20-2

在本实验中, 模拟量输出Ub为双极性, 当输入数字量小于80H时输出为负, 输入等于80H

时为0V，输入大于80H时输出为正。因而本实验中，DAC0832输入数字量只有2个(80H和FFH)，通过不同的延迟时间达到改变小电机转速的目的。

四、参考程序框图

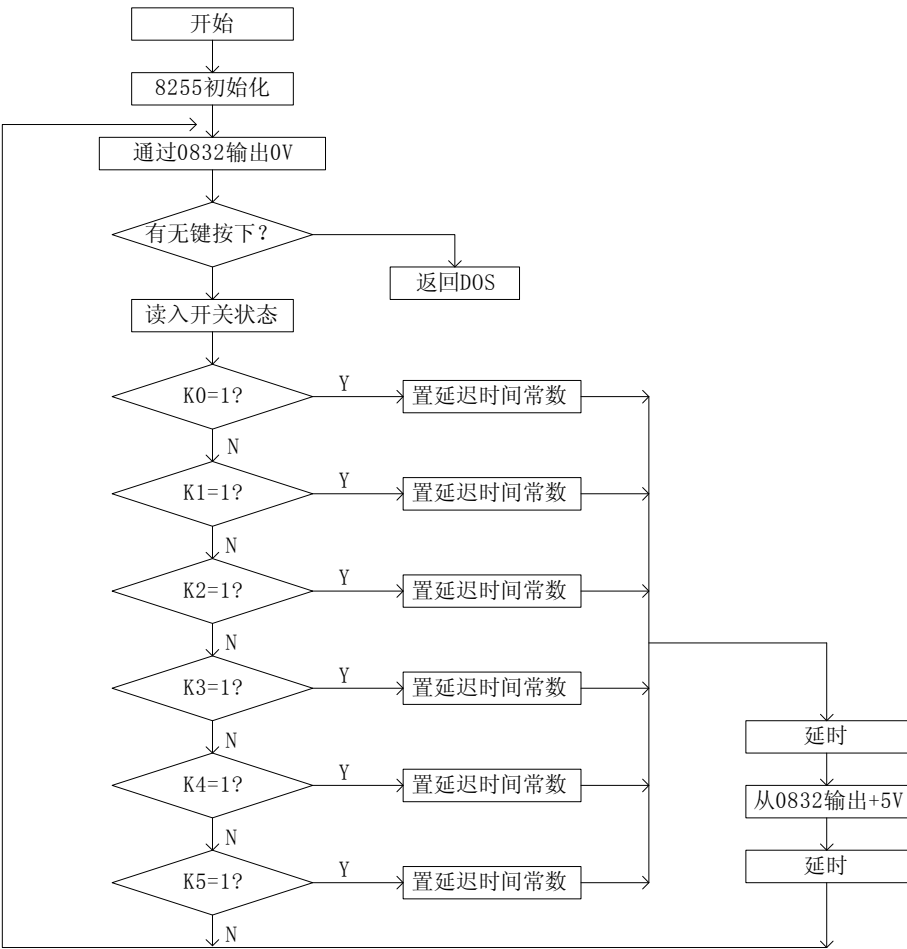


图19-3

五、参考程序：

1、 ZLDJ. ASM

```
data segment
port1      equ 290h
port2      equ 28bh
port3      equ 28ah
buf1       dw ?
buf2       dw ?
data ends
code segment
```

```

assume cs:code,ds:data
start:
    mov ax,data
    mov ds,ax
    mov dx,port2
    mov al,8bh
    out dx,al                ;8255 port c input
111:  mov al,80h
    mov dx,port1
    out dx,al                ;d/a output 0v
    push dx
    mov ah,06h
    mov dl,0ffh
    int 21h
    pop dx
    je intk                  ;not any key jmp intk
    mov ah,4ch
    int 21h                  ;exit to dos
intk:  mov dx,port3
    in al,dx                 ;read switch
    test al,01h
    jnz k0
    test al,02h
    jnz k1
    test al,04h
    jnz k2
    test al,08h
    jnz k3
    test al,10h
    jnz k4
    test al,20h
    jnz k5
    jmp 111
k0:   mov buf1,0400h
    mov buf2,0330h
delay: mov cx,buf1
delay1: loop delay1
    mov al,0ffh
    mov dx,port1

```

```

        out dx,al
        mov cx,buf2
delay2: loop delay2
        jmp l1l
k1:     mov buf1,0400h
        mov buf2,0400h
        jmp delay
k2:     mov buf1,0400h
        mov buf2,0500h
        jmp delay
k3:     mov buf1,0400h
        mov buf2,0600h
        jmp delay
k4:     mov buf1,0400h
        mov buf2,0700h
        jmp delay
k5:     mov buf1,0400h
        mov buf2,0800h
        jmp delay
code ends
end start

```

## 实验二十一 双色点阵发光二极管显示实验

### 一、实验目的

- 1、了解双色点阵LED显示器的基本原理。
- 2、掌握PC机控制双色点阵LED显示程序的设计方法。

### 二、实验原理

点阵LED显示器是将许多LED类似矩阵一样排列在一起组成的显示器件，双色点阵LED是在每一个点阵的位置上有红绿或红黄或红白两种不同颜色的发光二极管。当微机输出的控制信号使得点阵中有些LED发光，有些不发光，即可显示出特定的信息，包括汉字、图形等。车站广场由微机控制的点阵LED大屏幕广告宣传牌随处可见。

实验仪上设有一个共阳极8×8点阵的红绿两色LED显示器，其点阵结构如图所示。该点阵对外引出24条线，其中8条行线，8条红色列线，8条绿色列线。若使某一种颜色、某一个LED发光，只要将与其相连的行线加高电平，列线加低电平即可。

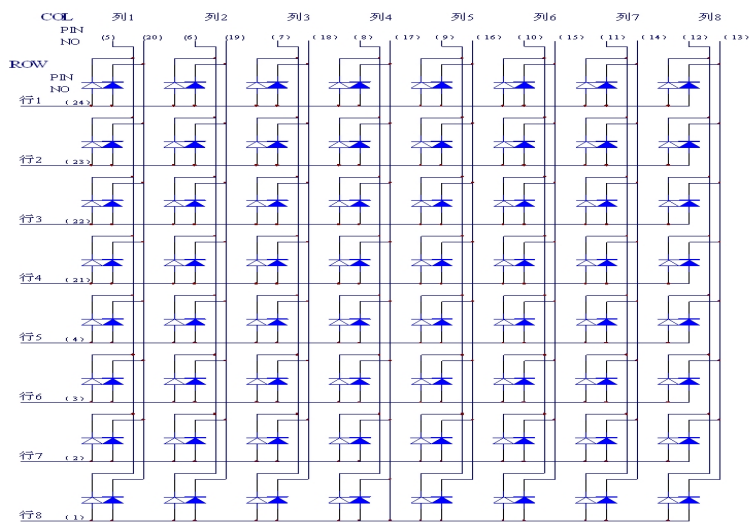


图4-21-1

例如欲显示汉字“年”，采用逐列循环发光。首先由“年”的点阵轮廓，确定点阵代码(如图所示)根据“年”的点阵代码，确定逐列循环发光的顺序如下：

- ① 行代码输出 44H； 红色列代码输 01H； 第一列2个红色LED发光。
- ② 行代码输出 54H； 红色列代码输 02H； 第二列3个红色LED发光。
- ③ 行代码输出 54H； 红色列代码输 04H； 第三列3个红色LED发光。
- ④ 行代码输出 7FH； 红色列代码输 08H； 第四列7个红色LED发光。
- ⑤ 行代码输出 54H； 红色列代码输 10H； 第五列3个红色LED发光。

⑥ 行代码输出 DCH； 红色列代码输 20H； 第六列5个红色LED发光。

⑦ 行代码输出 44H； 红色列代码输 40H； 第七列2个红色LED发光。

⑧ 行代码输出 24H； 红色列代码输 80H； 第八列2个红色LED发光。

在步骤①~⑧之间可插入几ms的延时，重复进行①~⑧即可在LED上稳定的显示出红色“年”字。若想显示绿色“年”，只需把红色列码改为绿色列码即可。

### 三、实验内容：

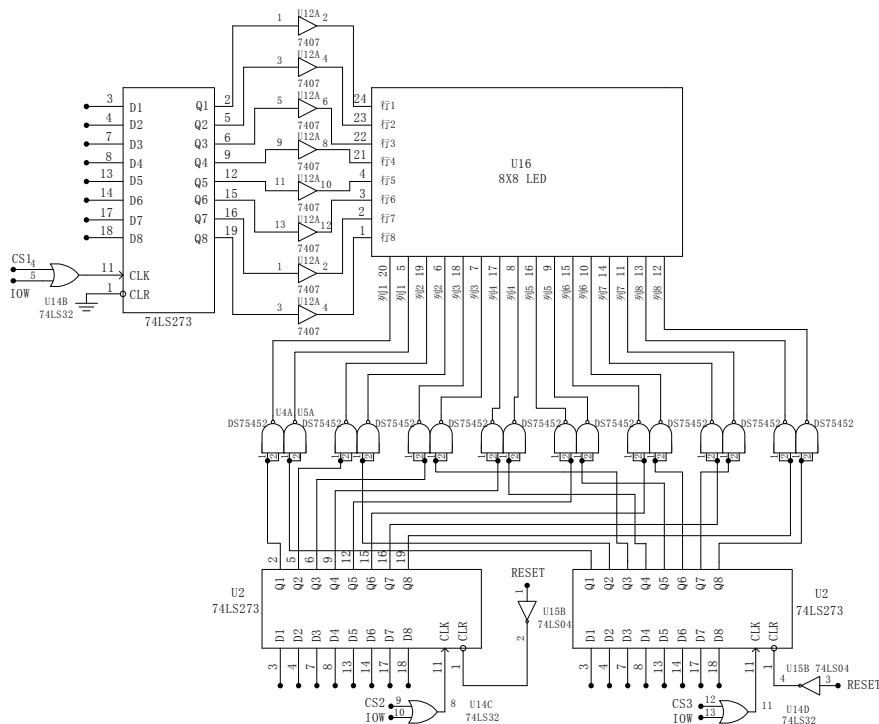


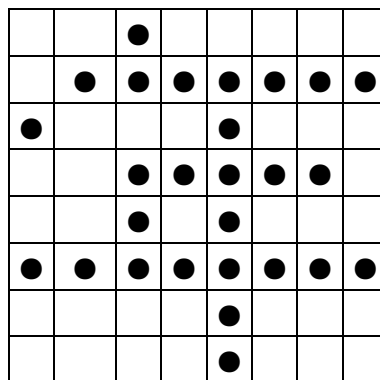
图4-21-2

1. 实验仪上的点阵LED及驱动电路如图3-24-2所示，点阵显示模块工作于总线模式时，行代码、红色列代码、绿色列代码各用一片74LS273锁存。行代码输出的数据通过行驱动器7407加至点阵的8条行线上，红和绿列代码的输出数据通过驱动器ULN2803反相后分别加至红和黄的列线上。行锁存器片选信号为“行选”，红色列锁存器片选信号为“红选”，绿色列锁存器片选信号为“绿选”。

2. 接线方法：行片选信号“行选”接 280H；红列片选信号“红选”接 288H；绿列片选信号“绿选”接 290H。点阵模块数据总线D7~D0接总线区的数据总线D7~D0。

3. 编程重复使LED点阵红色逐列点亮，再绿色逐列点亮，再红色逐行点亮，绿色逐行点亮。

4. 编程在LED上重复显示红色“年”和绿色“年”



5. 点阵显示模块工作于非总线模式，行代码、红代码、绿代码分别通过行驱动器7407和列反相驱动器ULN2803加到行线和列线上。如图4-21-3。

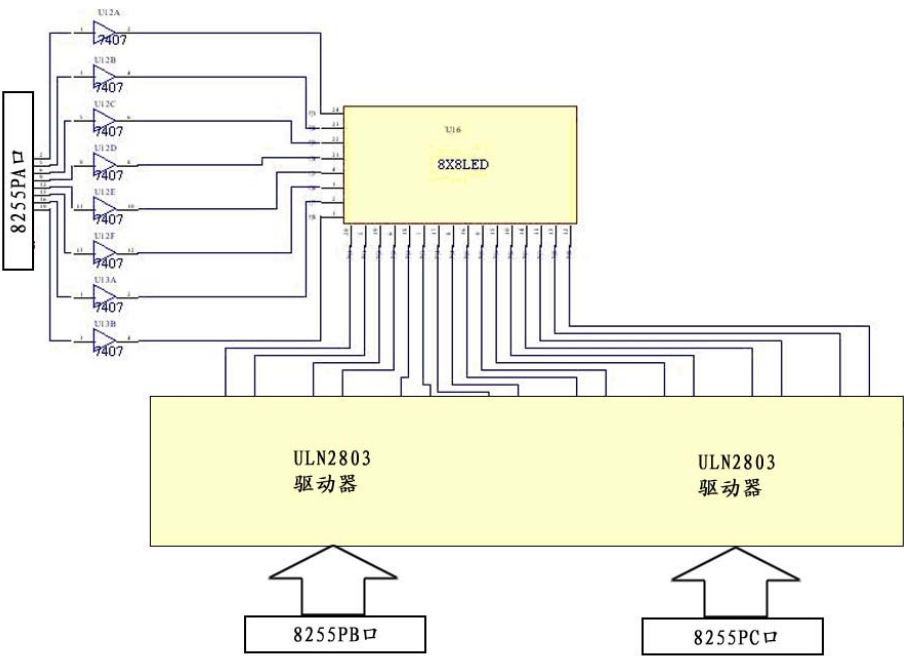


图4-21-3

6、接线：总线模式	D7~D0 /总线	接	D7~D0 /双色点阵
	Y0 /IO地址	接	行选 /双色点阵
	Y1 /IO地址	接	红选 /双色点阵
	Y2 /IO地址	接	绿选 /双色点阵
	IOW /总线	接	WR /双色点阵

双色点阵工作于“总线”模式（JCS4跳线短接总线边）。

#### 四、流程图

##### 1、逐行、逐列显示参考流程图1

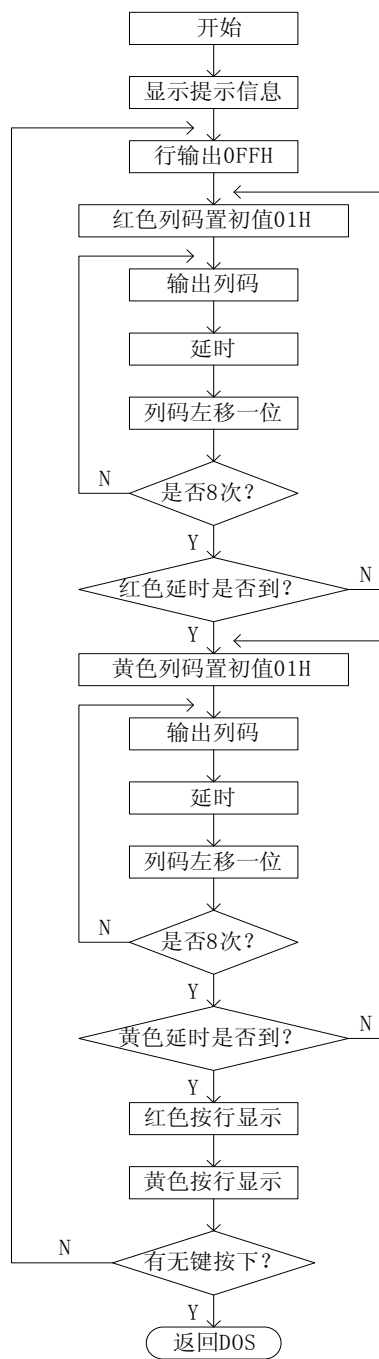
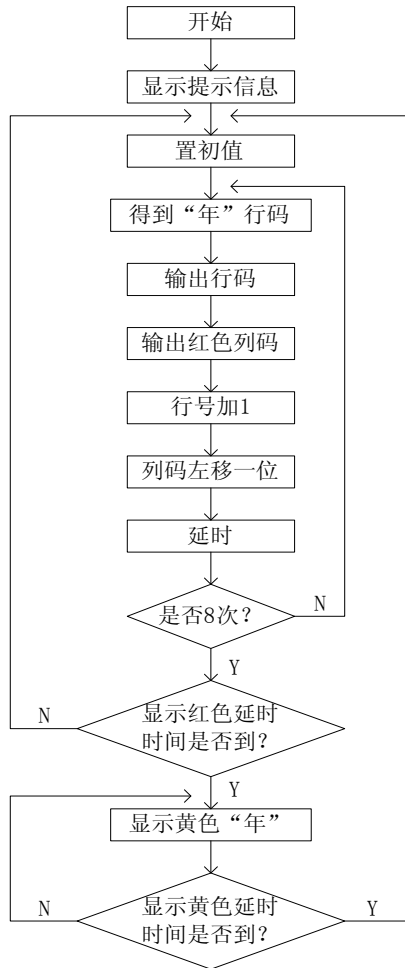


图22-3



## 2、显示“年”参考流程2



## 五、参考程序

### 1、逐行逐列显示参考程序 11588.asm

```

proth    equ 280h
protlr   equ 288h
protly   equ 290h
data segment
mess     db 'strike any key,return to dos!','0ah,0dh','$'
count    db 07
count1   dw 0000
buff     db 24h,22h,3bh,2ah,0feh,2ah,2ah,22h
data ends
code segment
assume cs:code,ds:data
start:
        mov ax,data

```

```

        mov ds,ax
        mov dx,offset mess
        mov ah,09
        int 21h                                ;显示提示信息
agn:    mov al,0ffh
        mov dx,proth
        out dx,al
        mov ah,01
        mov cx,0008h
agn1:   shl ah,01
        mov dx,protlr
        mov al,ah
        out dx,al                                ;红行亮
        push cx
        mov cx,0030h
d5:     call delay1
        loop d5
        pop cx
        loop agn1
        mov ah,01
        int 16h
        jnz a2
        mov ah,01
        mov cx,0008h
agn2:   mov dx,protly
        mov al,ah
        out dx,al
        push cx
        mov cx,0030h                                ;绿行亮
d4:     call delay1
        loop d4
        pop cx
        shl ah,01
        loop agn2
        mov ah,01
        int 16h
        jnz a2
        mov al,0ffh
        mov dx,protlr

```

```

        out dx,al                ;红列亮
        mov ah,01
        mov cx,0008h
agn3:   mov dx,proth
        mov al,ah
        out dx,al
        push cx
        mov cx,0030h
d2:     call delay1
        loop d2
        pop cx
        shl ah,01
        loop agn3
        mov al,00h
        mov dx,protlr
        out dx,al
        mov ah,01
        int 16h
        jnz a2
        mov al,0ffh
        mov dx,protly
        out dx,al                ;绿列亮
        mov ah,01
        mov cx,0008h
agn4:   mov dx,proth
        mov al,ah
        out dx,al
        push cx
        mov cx,0030h
d1:     call delay1
        loop d1
        pop cx
        shl ah,01
        loop agn4
        mov ah,01
        int 16h
        jnz a2
        jmp agn
delay1 proc near                ;延迟子程序

```

```

        push cx
        mov cx, 4000h
ccc:    loop ccc
        pop cx
        ret
delay1 endp
a2:     mov ah, 4ch          ;返回
        int 21h
code ends
end start

```

## 2、显示“年”参考程序 11588-1.ASM

```

proth equ 280h
protlr equ 288h
protly equ 290h
data segment
mess db 'Strike any key, return to DOS!', 0AH, 0DH, '$'
minl db 00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
count db 0
buff db 44h, 54h, 54h, 7fh, 54h, 0dch, 44h, 24h
data ends
code segment
        assume cs:code, ds:data
start:  mov ax, data
        mov ds, ax
        mov dx, offset mess
        mov ah, 09
        int 21h          ;显示提示信息
agn:    mov cx, 80h
d2:     mov ah, 01h
        push cx
        mov cx, 0008h
        mov si, offset minl
next:   mov al, [si]
        mov bx, offset buff
        xlat             ;得到第一行码
        mov dx, proth
        out dx, al
        mov al, ah

```

```

        mov dx,protlr
        out dx,al           ;显示第一行红
        shl ah,01
        inc si
        push cx
        mov cx,8fffh
delay2: loop delay2        ;延时
        pop cx
        loop next
        pop cx
        call delay
        loop d2
        mov al,00
        mov dx,protlr
        out dx,al
        mov ah,01          ;有无键按下
        int 16h
        jnz a2
agn1:   mov cx,80h          ;agn1为显示绿色
d1:     mov si,offset minl
        mov ah,01
        push cx
        mov cx,0008h
next1:  mov al,[si]
        mov bx,offset buff
        xlat
        mov dx,proth
        out dx,al
        mov al,ah
        mov dx,protly
        out dx,al
        shl ah,01
        inc si
        push cx
        mov cx,8fffh
delay1: loop delay1
        pop cx
        loop next1
        pop cx

```

```

        call delay
        loop d1
        mov al,00
        mov dx,protly
        out dx,al
        mov ah,01
        int 16h
        jnz a2
        jmp agn          ;绿色红色交替显示
delay proc near          ;延迟子程序
        push cx
        mov cx,0ffffh
ccc:    loop ccc
        pop cx
        ret
delay endp
a2:     mov ah,4ch
        int 21h          ;返回
        code ends
        end start

```

## 实验二十二 128X64字符图形液晶显示实验

### 一、实验目的

掌握LCD图形显示模块接口方法，掌握LCD显示模块显示汉字与字符的编程方法。

### 二、实验原理和内容

1、使用8255与128X64LCD显示模块连接，编程显示汉字字符串“32位微机教学实验系统正在演示中”。

2、接线：PA7~PA0 /8255 接 D7~D0 /LCD

PC0 /8255 接 D/I /LCD

PC1 /8255 接 RW /LCD

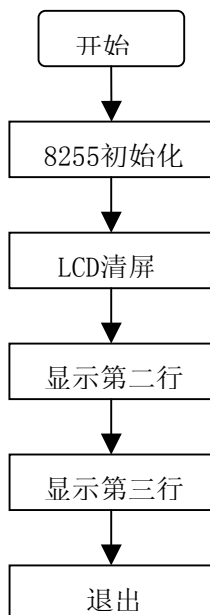
PC2 /8255 接 E /LCD

LCD字符图形液晶工作于并行模式

### 三、实验提示

实验系统LCD字符图形液晶为ST7920A控制器，LCD字符图形液晶详细资料请查阅附录中的LCD资料。

### 四、流程图



## 五、参考程序 LCD.ASM

;LCD.ASM

I08255A equ 288h

I08255C equ 28Ah

I08255KZ equ 28bh

DATA SEGMENT

HZ\_TAB DW 0A3B3H, 0A3B2H, 0CEBBH, 0CEA2H, 0BBFAH, 0BDCCH, 0D1A7H, 0CAB5H

DW 0D1E9H, 0CFB5H, 0CDB3H, 0D5FDH, 0D4DAH, 0D1DDH, 0CABEH, 0D6D0H

HZ\_ADR DB ? ;存放显示行起始端口地址

DATA ENDS

code segment

assume cs:code,ds:data

START: MOV AX, DATA

MOV DS, AX

MOV DX, I08255KZ

MOV AL, 80H

OUT DX, AL ;8255初始化

CALL CLEAR ;LCD 清除

LEA BX, HZ\_TAB

MOV CH, 2 ;显示第2行信息

CALL LCD\_DISP

LEA BX, HZ\_TAB

MOV CH, 3 ; 显示第3行信息

CALL LCD\_DISP

mov ah, 4ch ;退出

int 21h

CLEAR PROC

MOV AL, 0CH

MOV DX, I08255A

OUT DX, AL ;设置CLEAR命令

CALL CMD\_SETUP ;启动LCD执行命令

RET

CLEAR ENDP

LCD\_DISP PROC

LEA BX, HZ\_TAB

CMP CH, 2

JZ DISP\_SEC

MOV BYTE PTR HZ\_ADR, 88H ;第三行起始端口地址

ADD BX, 16 ;指向第二行信息

JMP next

DISP\_SEC: MOV BYTE PTR HZ\_ADR, 90H

next: mov cl, 8



```

continue:    push cx
             MOV AL, HZ_ADR
             MOV DX, I08255A
             OUT DX, AL
             CALL CMD_SETUP           ;设定DDRAM地址命令
             MOV AX, [BX]
             PUSH AX
             MOV AL, AH               ;先送汉字编码高位
             MOV DX, I08255A
             OUT DX, AL
             CALL DATA_SETUP        ;输出汉字编码高字节
             CALL DELAY              ;延迟
             POP AX
             MOV DX, I08255A
             OUT DX, AL
             CALL DATA_SETUP        ;输出汉字编码低字节
             CALL DELAY
             INC BX
             INC BX                   ;修改显示内码缓冲区指针
             INC BYTE PTR HZ_ADR    ;修改LCD显示端口地址
             POP CX
             DEC CL
             JNZ CONTINUE
             RET

LCD_DISP    ENDP

CMD_SETUP   PROC
             MOV DX, I08255C         ;指向8255端口控制端口
             NOP
             MOV AL, 00000000B       ;PC1置0, pc0置0 (LCD I端=0, W端=0)
             OUT DX, AL
             call delay
             NOP
             MOV AL, 00000100B       ;PC2置1 (LCD E端=1)
             OUT DX, AL
             NOP
             call delay
             MOV AL, 00000000B       ;PC2置0, (LCD E端置0)
             OUT DX, AL
             call delay

             RET
CMD_SETUP   ENDP

DATA_SETUP  PROC

```

```

MOV DX, I08255C           ;指向8255控制端口
MOV AL, 00000001B        ;PC1置0, PC0=1 (LCD I端=1)
OUT DX, AL
NOP
call delay
MOV AL, 00000101B        ;PC2置1 (LCD E端=1)
OUT DX, AL
NOP
call delay
MOV AL, 00000001B        ;PC2置0, (LCD E端=0)
OUT DX, AL
NOP
call delay
RET
DATA_SETUP               ENDP

DELAY                   PROC
push cx
push dx
MOV CX, 0Ffffh
x1: loop x1
pop dx
pop cx
RET
DELAY                   ENDP

code ends
end start

```

## 实验二十三 集成电路测试

## 一、实验目的

通过对四与非门(74LS00)集成电路芯片的测试,了解测试一般数字集成电路方法,进一步熟悉可编程并行接口8255 的使用。

## 二、实验原理和内容

1、按图4-23-1连接电路，其中74LS00插在通用插座上。

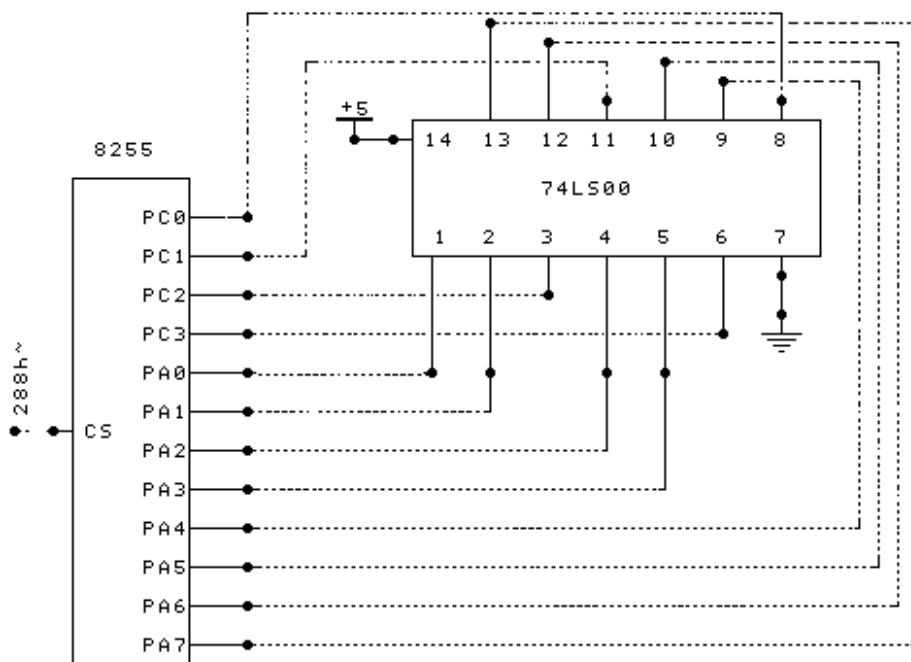


图4-23-1 集成电路测试电路图

## 2、编程提示:

将8255设置在方式0下工作，使A口输出，C口输入，先后通过A口分别给每一个门电路送入四个测试信号(00、01、10、11)，相应从C口读出每一个门电路的输出结果，与正常值(1、1、1、0)进行比较，若相等，则此芯片好，否则为坏。

3、接线：按图4-23-1接线（图中虚线为实验所需接线）

### 三、参考流程图 (见图4-23-2)

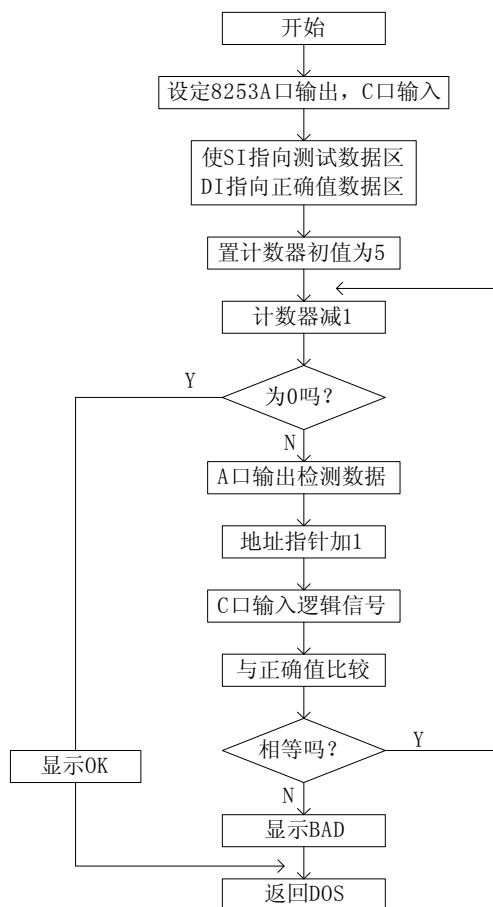


图4-23-2 集成电路测试流程图

#### 四、参考程序:

##### 1、JC. ASM

```

data    segment
io8255a    equ 288h
io8255b    equ 28ah
io8255c    equ 28bh
se         db 00000000b           ;检测时发送的数据
          db 01010101b
          db 10101010b
          db 11111111b
ac0        db 00001111b           ;74LS00正确时检测时接收的数据
          db 00001111b
          db 00001111b
          db 00000000b
outbuf     db ' THE CHIP IS OK', 07h, 0ah, 0dh, '$'
news       db ' THE CHIP IS BAD', 07h, 0ah, 0dh, '$'

```

```

data ends
code segment
assume cs:code, ds:code
start:
    mov ax, data
    mov ds, ax
    mov dx, io8255c           ;对8255进行初始化编程
    mov al, 89h              ;使A口输出, C口输入
    out dx, al
    mov di, offset ac0       ;DI中存放接收数据的缓冲区首址
    mov si, offset se        ;SI中存放发收数据的缓冲区首址
    mov cx, 05h              ;发送四个字节
again:  dec cx
        jz exit              ;如果四个数值都相等, 则显示提示信息
        mov dx, io8255a
        mov al, [si]
        mov bl, [di]
        out dx, al           ;发送数据
        inc si
        inc di
        mov dx, io8255b
        in al, dx             ;读芯片的逻辑输出
        and al, 0fh
        cmp al, bl
        je again             ;若正确就继续
error:  mov dx, offset news   ;若有错, 芯片有问题
        mov ah, 09h           ;显示错误的提示信息
        int 21h
        jmp ppp
exit:   mov dx, offset outbuf ;显示正确的提示信息
        mov ah, 09h
        int 21h
ppp:    mov ah, 4ch           ;返回
        int 21h
code ends
end start

```

## 实验二十四 电子琴

### 一、实验目的

- 1、通过8254产生不同的频率信号，使PC机成为简易电子琴。
- 2、了解利用8255和8254产生音乐的基本方法。

### 二、实验原理和内容

1、实验电路如图4-24-1，8254的CLK0接1MHz时钟，GATE0接8255的PC1，OUT0和8255的PC0接到与门的两个输入端，与门输出Y连接喇叭，编程使计算机的数字键1、2、3、4、5、6、7作为电子琴按键，按下即发出相应的音阶。

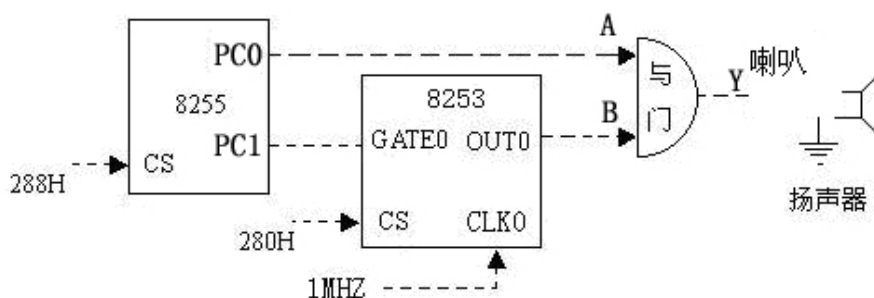


图4-24-1 电子琴电路

- 2、接线：
- |           |   |             |
|-----------|---|-------------|
| Y1 /IO地址  | 接 | CS /8255    |
| Y0 /IO地址  | 接 | CS /8254    |
| 1M时钟      | 接 | CLK0 /8254  |
| PC1 /8255 | 接 | GATE0 /8254 |
| A/与门      | 接 | OUT0 /8254  |
| B/与门      | 接 | PC0 /8255   |
| Y/与门      | 接 | 喇叭          |

### 三、编程提示：

1、利用8255的PC0口来施加控制信号给与门，用来控制扬声器的开关状态。再利用设置不同的计数值，使8254产生不同频率的波形，使扬声器产生不同频率的音调，达到类似与音阶的高低音变换。对于音乐，每个音阶都有确定的频率。

各音阶标称频率值：

音 阶	1	2	3	4	5	6	7	1 <sub>·</sub>
低频率(单位:Hz)	262	294	330	347	392	440	494	524
高频率(单位:Hz)	524	588	660	698	784	880	988	1048

### 四、参考流程图 （见图4-24-2）

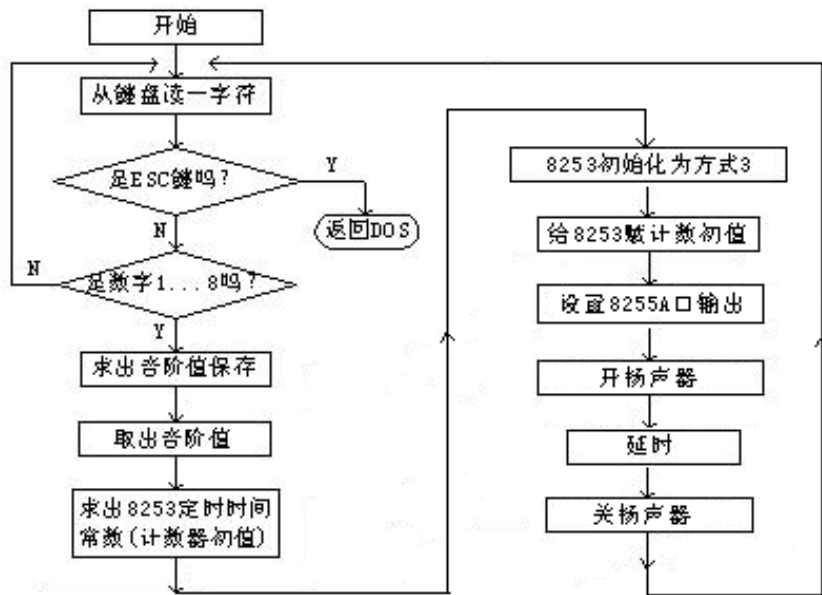


图4-24-2 主程序

## 五、参考程序：

### 1、DZQ.ASM

```

;*****
;*          电子琴实验          *
;*****
;此实验接线如下：
;8254 CLK0接1MHZ时钟，GATE0接8255的PC1，OUT0接与门输入端1，CS接280H~287H，
;8255 PC0接与门输入端2，CS接288H~28FH，
;与门输出端接 'LB' . JD3用跳线端子接好

```

```

data segment
io8255c      equ 28ah
io8255ctl    equ 28bh
io8254a      equ 280h
io8254b      equ 283h
table dw 524, 588, 660, 698, 784, 880, 988, 1048; 高音的
;table dw 262, 294, 330, 347, 392, 440, 494, 524; 低音的
msg db 'Press 1, 2, 3, 4, 5, 6, 7, 8, ESC:', 0dh, 0ah, '$'
data ends

```

```

code segment
assume cs:code, ds:data
start:
    mov ax, data
    mov ds, ax

    mov dx, offset msg

```

```

    mov ah,9
    int 21h                ;显示提示信息
sing:
    mov ah,7
    int 21h                ;从键盘接收字符,不回显
    cmp al,1bh
    je finish              ;若为ESC键,则转finish
    cmp al,'1'
    jl sing
    cmp al,'8'
    jg sing                ;若不在'1'-'8'之间转sing

    sub al,31h
    shl al,1               ;转为查表偏移量
    mov bl,al              ;保存偏移到bx
    mov bh,0

    mov ax,4240H           ;计数初值 = 1000000 / 频率, 保存到AX
    mov dx,0FH
    div word ptr[table+bx]
    mov bx,ax

    mov dx,io8254b         ;设置8254计时器0方式3, 先读写低字节, 再读写高字节
    mov al,00110110B
    out dx,al

    mov dx,io8254a
    mov ax,bx
    out dx,al              ;写计数初值低字节

    mov al,ah
    out dx,al              ;写计数初值高字节

    mov dx,io8255ctl       ;设置8255 C口输出
    mov al,10000000B
    out dx,al

    mov dx,io8255c
    mov al,03h
    out dx,al              ;置PC1、PC0 = 11(开扬声器)
    call delay             ;延时
    mov al,0h
    out dx,al              ;置PC1、PC0 = 00(关扬声器)

    jmp sing

```



```

finish:
    mov ax, 4c00h
    int 21h

delay proc near           ;延时子程序
    push cx
    push ax
    mov ax, 15
x1: mov cx, 0ffffh
x2: dec cx
    jnz x2
    dec ax
    jnz x1
    pop ax
    pop cx
    ret
delay endp
code ends
end start

```