



80X86 微机原理及接口技术 实验指导书

卓然 编著



2015-3-1

南京航空航天大学
电工电子实验教学中心

目 录

序	2
第一章 TD-PIT++实验系统简介	3
1. 概述	3
2. 系统总线电路单元	4
3. 接口实验单元	6
第二章 32 位微机原理软件实验	16
实验一 四则运算	16
实验二 数据统计	19
实验三 代码转换	22
实验四 数据块移动	27
实验五 描述符和描述符表实验（选做）	29
实验六 局部描述符表实验（选做）	33
第三章 32 位微机接口硬件实验	37
实验一 地址译码电路与 I/O 接口	37
实验二 8254 定时/计数器	44
实验三 8259 中断控制器	50
实验四 8255 并口控制器	58
实验五 A/D 与 D/A 转换实验	66
第四章 32 位微机接口课程设计	73
课程设计一 数据采集系统一	73
课程设计二 数据采集系统二（查询法）	77
课程设计三 数据采集系统三（中断法）	81
课程设计四 信号发生器	85
课程设计五 交通灯实时控制系统设计	93
课程设计六 步进电机控制系统设计	99
附录 A 常用 DOS 系统功能（INT 21H）	105
附录 B TDPIT 集成操作软件使用说明	106

序

本教材是为南航“微机原理”及“计算机硬件技术基础”两门课程编写的配套实验指导书，包括这两门课程的实验及课程设计的相关内容。

实验，是科学研究的基本方法之一，不能将其理解为对理论知识的加强及延伸，恰恰相反，自然科学理论是从大量科学实验中归纳而来。实验更是同学们获取第一手知识的重要手段，实验中要充分发扬打破沙锅问到底的精神。

笔者根据近年来对本课程的教学经验，对王位喜老师编写的实验教材做了大幅修订，丰富了原教材的内容，也更正了一些错误。

本教材分为四章，做简单介绍如下：

第一章介绍了实验系统的构成原理，建议同学们在开始实验之前先自行阅读第一章内容，对实验系统有一个初步的认识；

第二章内容为软件实验，通过软件实验，同学们可以更快的熟悉80X86汇编语言的编程及调试方法，学有余力的同学还可以参考选做实验；

第三章内容为硬件接口实验，基础实验将结合理论课的讲授内容，并对应实验平台上的各个单元展开。选做实验为综合型和设计型实验，为学有余力的同学提高之用；

第四部分内容为课程设计，为课程设计环节提供指导；

附录B为实验用到的集成开发环境使用说明，建议同学们在实验开始之前自行阅读，也可在遇到问题时进行查阅。

鉴于作者水平有限，书中必然存在许多不妥之处，甚至错误，殷切希望有关专家和广大读者批评指正，以便再版时修改。

编者

2015年3月

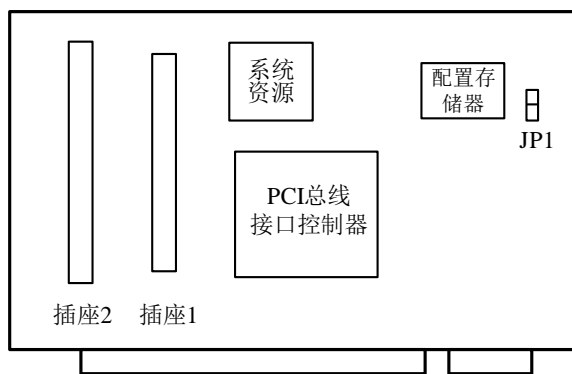


图 1-1A PCI 总线扩展卡结构

1.2TD-PIT++实验平台

在 TD-PIT++实验平台上,电路结构主要分两部分：系统总线单元电路和实验单元电路,图 1-2 是 32 位微机接口实验的主要操作平台。

先熟悉实验平台的结构以及各个单元，有助于同学们更好的理解后续实验内容。

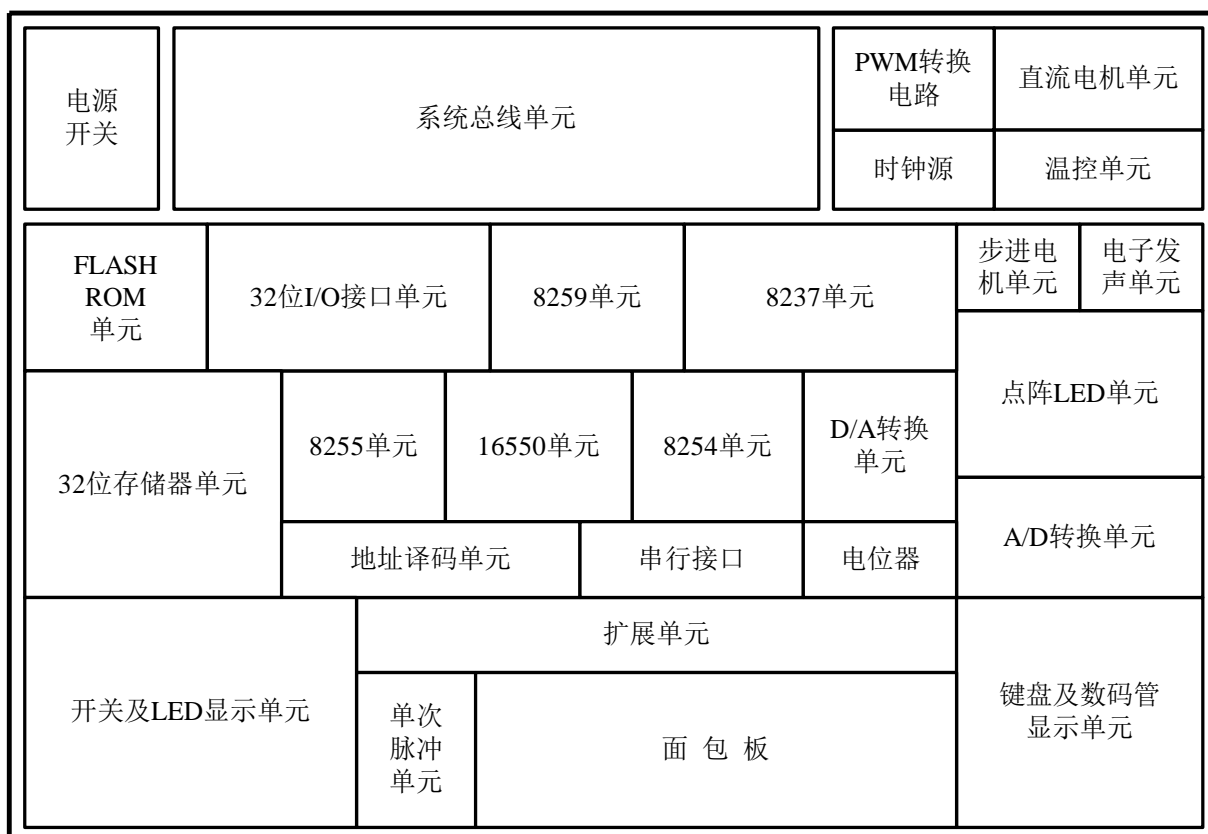


图 1-2 实验平台结构

2. 系统总线电路单元

总线单元实现了面向80x86 微机系统的32 位系统总线，符合80x86 总线时序标的接口电路均可以直接连接到该总线上。

系统总线单元分为数据总线、地址总线、控制总线三部分，32位的数据总线编号为XD0~XD31，32位的地址总线编号为XA0~XA31（其中XA0、XA1两条地址线保留，实验箱上并未留出接口），其余为控制总线。

总线信号说明如表1-1 所示。

表1-1 总线信号

信号名称	含义
XD[31:0]	32 位数据总线
XA[31:2]	32 位地址总线
XMER、XMEW、XIOR、XIOW	存储器读写信号、I/O 读写信号
IOY0 、IOY1、 IOY2、 IOY3	I/O 空间片选信号
MY0、MY1、MY2、MY3	存储器空间片选信号
BE0、BE1、BE2、BE3	32 位数据字节使能信号
HOLD、HLDA	总线保持请求和总线保持响应信号
INTR	中断请求信号(上升沿有效)
CLK	系统时钟 CLK = 1.041667MHz
RST、RST#	系统复位信号

注：# 号表示该信号低电平有效

实验系统向PC 机申请了接口实验所需的配置资源。其中包括16MB 的存储地址空间、255字节的I/O 地址空间和一个中断请求线。中断请求线是映射到PC 机内15 个中断线中的一个。系统总线单元将地址空间进行了译码，各提供4 个片选信号，片选信号同偏移地址空间对应关系如表1-2 所示。

表1-2片选信号同偏移地址空间对应关系

片选信号	偏移地址范围	片选信号	偏移地址范围
IOY0	3000~303FH	MY0	000000~3FFFFFFH
IOY1	3040~307FH	MY1	400000~7FFFFFFH
IOY2	3080~30BFH	MY2	800000~BFFFFFFH
IOY3	30C0~30FFH	MY3	C00000~FFFFFFH

用PC 机分配的I/O 或存储器空间始地址加上这个偏移地址，就是实验系统中端口占用的实际地址，I/O 和或存储器地址电原理如图1-2A所示。PC 机分配的起始地址可以在Tdpit 软件中查看或由实验系统附带的配置资源检查程序CHECK.EXE 获得。

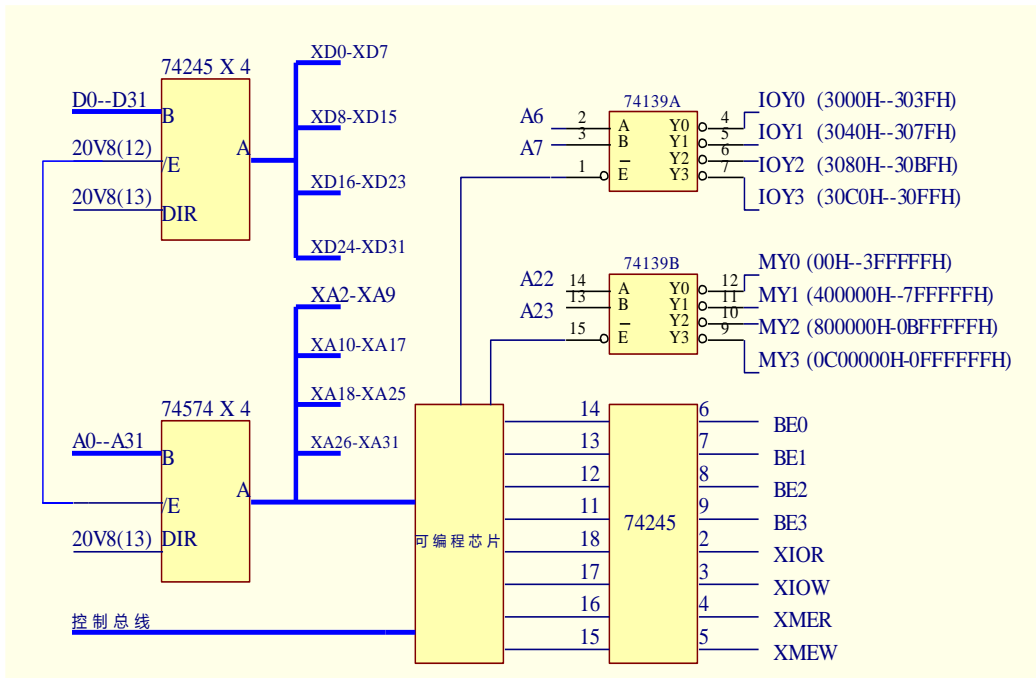


图1-2A I/O和存储器地址电原理图

3. 接口实验单元

1. 地址译码单元

该单元提供一片开放的译码器74LS138，用于学习地址译码方法。其线路连接如图1-3所示。

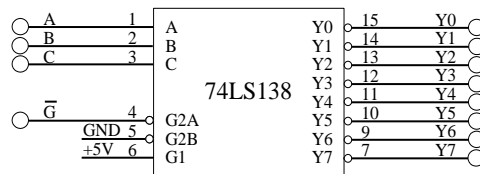


图 1-3 地址译码单元

2. 32 位I/O 接口单元

该单元通过4 片三态缓冲器和4 片锁存器组成32 位的I/O 接口，并根据32 位总线时序设计了译码电路，可以8/16/32 位不同字节宽度来访问该接口。用于学习8 位和32 位I/O 接口设计及编程。其线路连接如图1-4 所示。

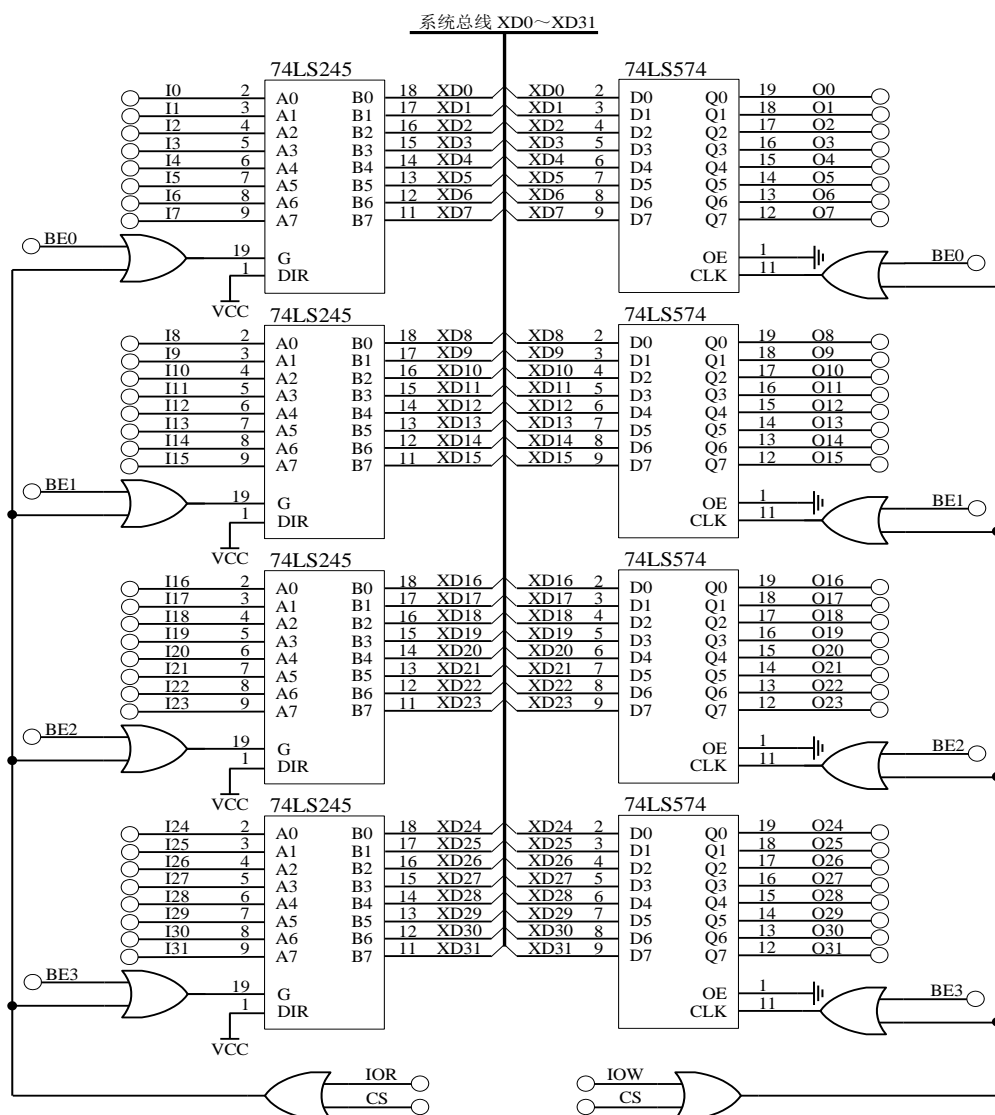


图 1-4 32 位 I/O 接口单元

3. 32 位存储器单元

该单元提供32 位存储器及其连接电路，并针对32 位系统总线提供了存储器译码电路，可以任意完成8 位、16 位及32 位不同字节宽度的存储器操作。其线路连接如图1-5 所示。

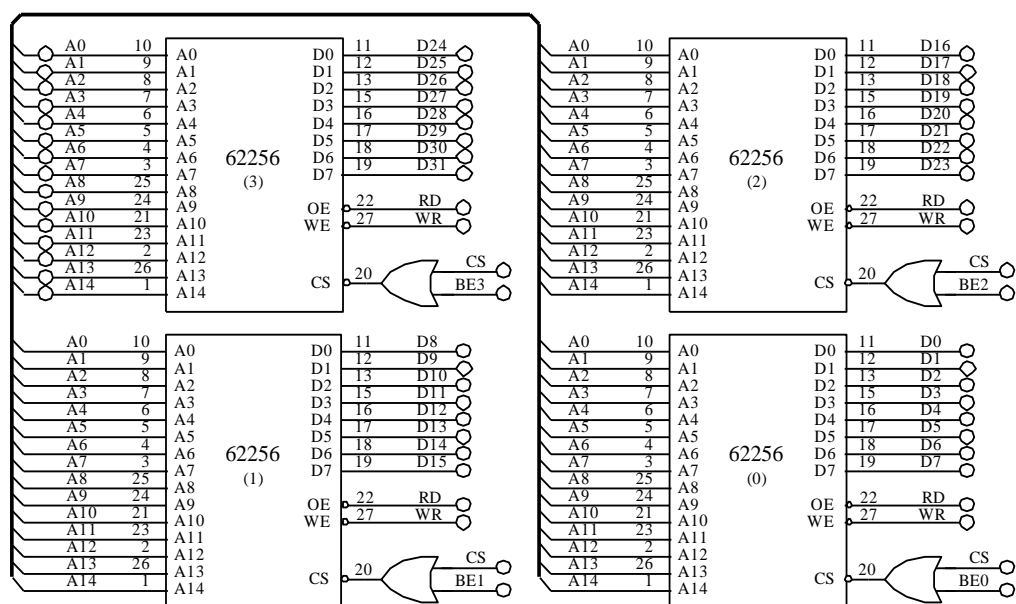


图 1-5 32 位存储器单元

4. FLASH ROM 单元

该单元提供一片开放的FLASH 存储器，用于学习FLASH 存储器的编程操作方法。其线路连接如图1-6 所示。

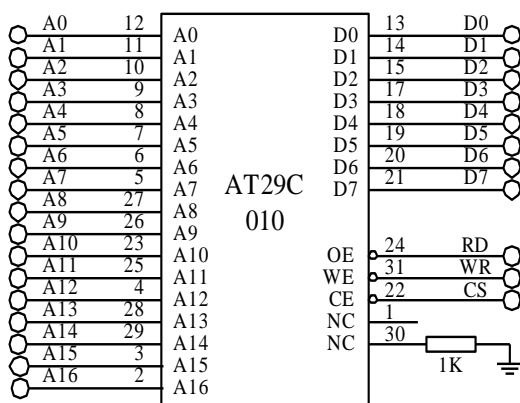


图 1-6 FLASH ROM 单元

5. 8259 单元

该单元提供中断控制器8259 的连接电路，用于学习中断控制器的操作方法。其线路连接如图 1-7 所示。

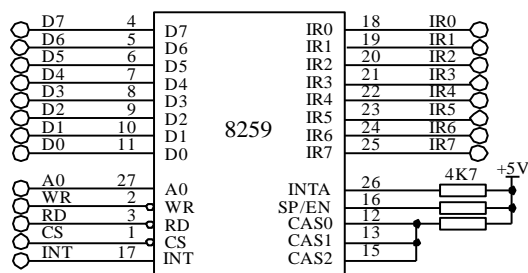


图 1-7 8259 单元

6. 8237 单元

该单元提供DMA 控制器8237 的连接电路，用于学习DMA 传送应用编程方法。其线路连接如图1-8 所示。

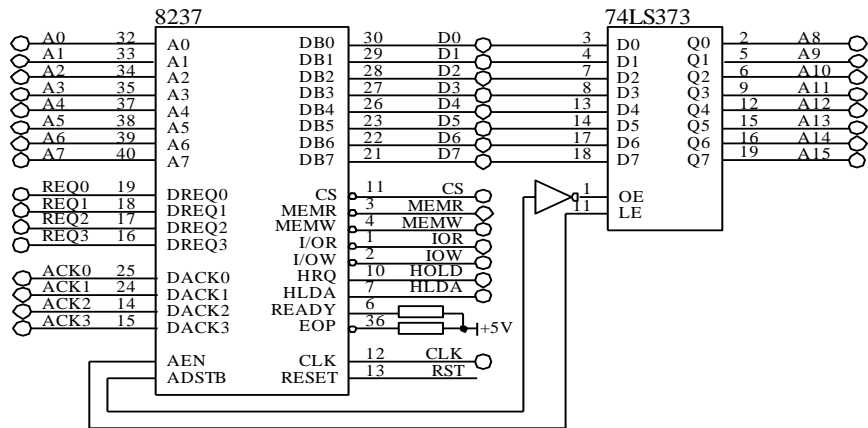


图 1-8 8237 单元

7. 8255 单元

该单元提供一片开放的并口控制器8255，用于学习并行接口8255 的编程方法。其线路连接如图1-9 所示。

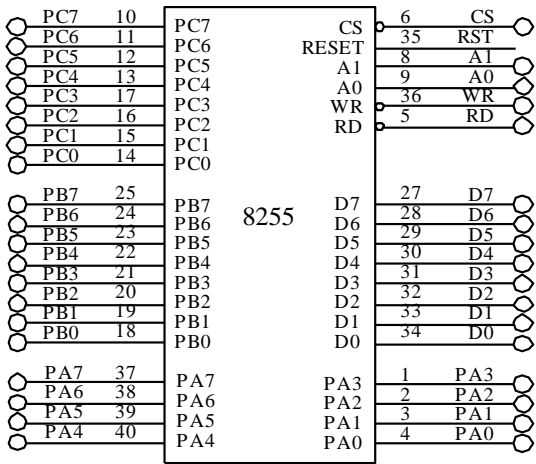


图 1-9 8255 单元

8. 8251 单元

该单元提供串行控制器8251 的连接电路，用于学习串行通讯编程方法。其线路连接如图1-10 所示。

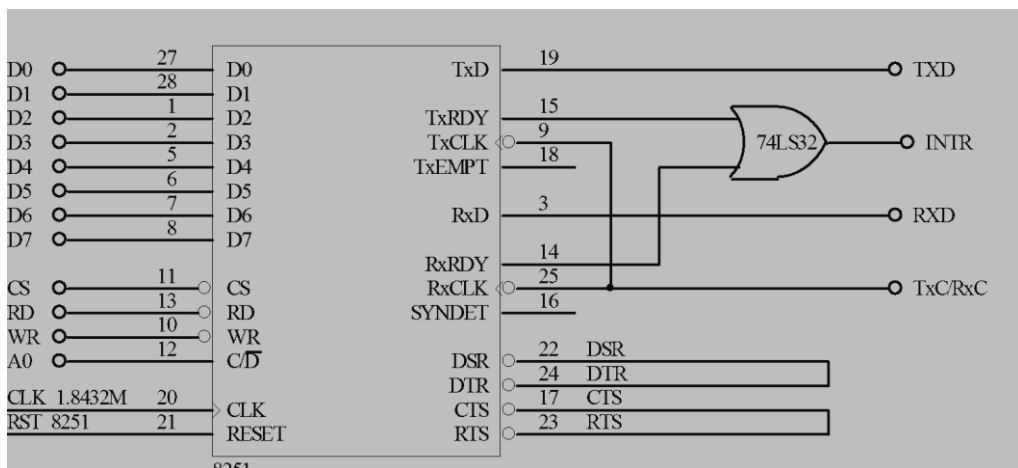


图 1-10 8251 单元

9. 串行接口单元

该单元提供用来将串行通讯信号引出到实验箱体后侧的接口插座，包含USB 和RS232 接口。使用USB 接口插座时将VDD、D-、D+和GND 四个信号连接到标有USB 的插座上。使用RS232串口时将TXD、RXD 信号连接到标有RS-232 的插座上。其电路结构如图1-11 所示。

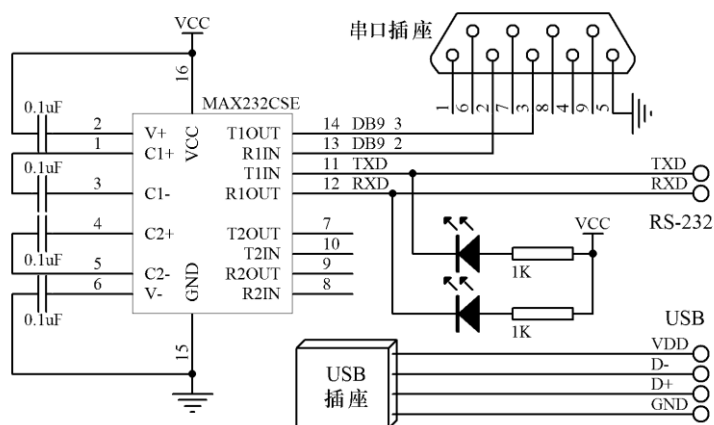


图 1-11 串行接口单元

10. 8254 单元

该单元提供一片开放的定时/计数器8254，用于学习定时/计数器的应用编程方法。其线路连接如图1-12 所示。

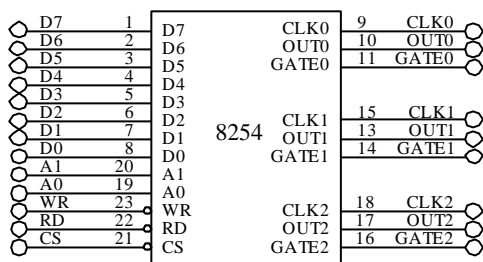


图 1-12 8254 单元

11. A/D 转换单元

该单元提供模/数转换器ADC0809 的连接电路，用于学习A/D 转换原理及编程操作方法。其线路连如图1-13 所示。

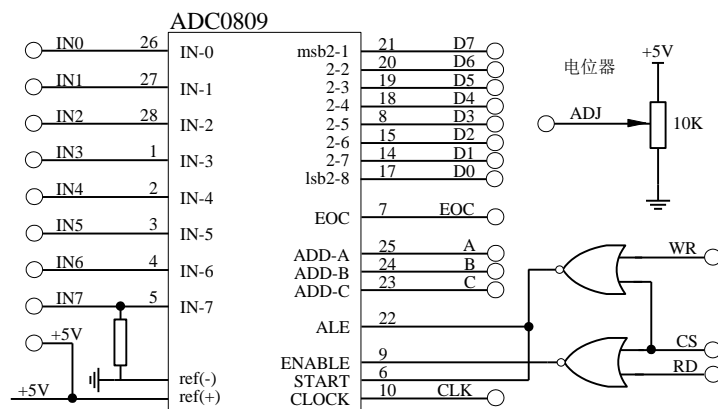


图 1-13 A/D 转换单元

12. D/A 转换单元

该单元提供数/模转换器DAC0832 的连接电路，用于学习D/A 转换原理及编程操作方法。其线路连接1-14 所示。

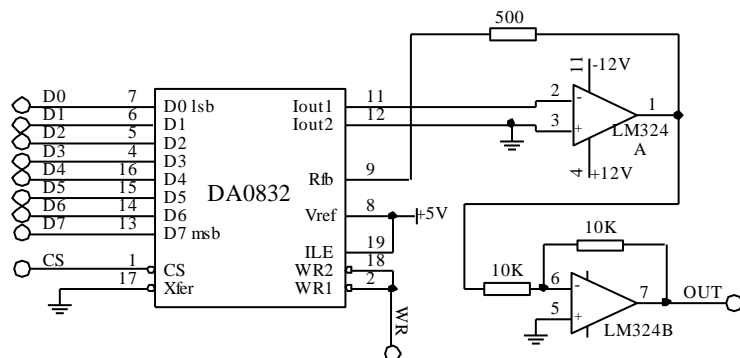


图 1-14 D/A 转换单元

13. 电子发声单元

该单元提供一个微型扬声器，控制和驱动电路已经连接好。在控制输入端输入一定频率的波形信号即可发声。其线路连接电路如图1-15 所示。

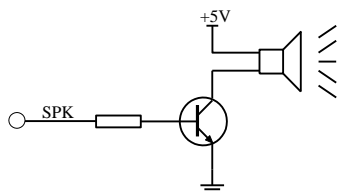


图 1-15 电子发声单元

14. 键盘及数码管显示单元

该单元提供4×4 的小键盘矩阵及6 位七段数码管，电路连接为扫描电路形式。其线路连接如图1-16 所示。

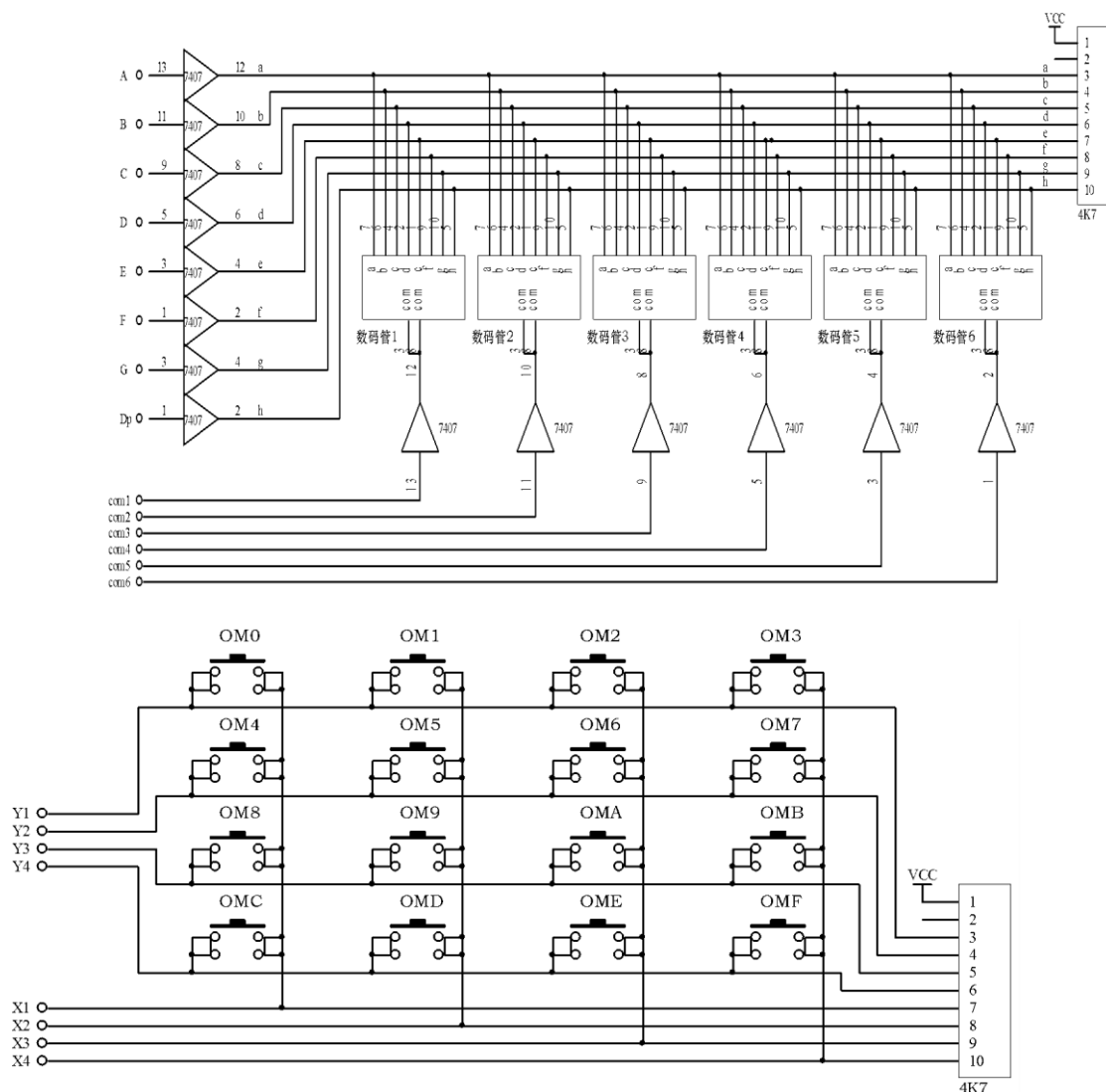


图 1-16 键盘及数码管显示单元

15. 点阵LED 显示单元

该单元使用4 片8×8 的点阵LED 构成了一个16×16 点阵显示模块，点阵LED 的行列控制已经连接好。行控制为R0~R15，列控制为L0~L15。其线路连接如图1-17 所示。

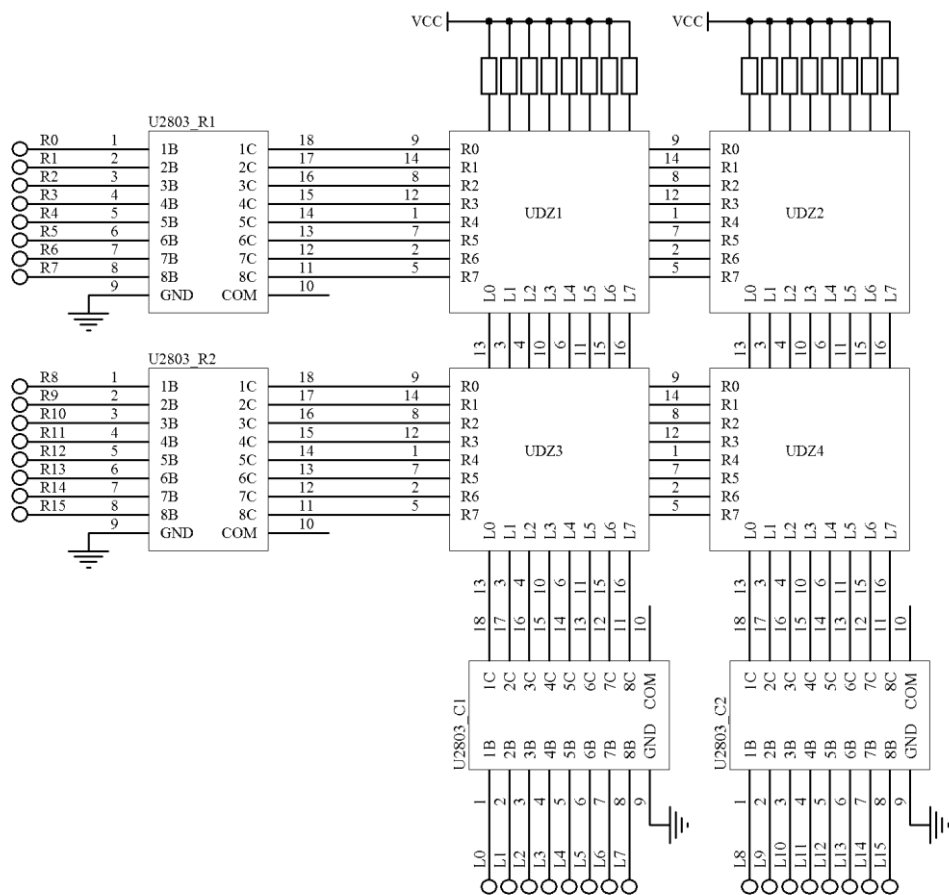


图 1-17 点阵 LED 显示单元

16. 驱动电路和直流电机单元

这两个单元由ULN2803 驱动芯片、一台DC12V 直流电机及霍尔测速电路构成，N 为一组反相驱动信号输入端。其线路连接如图1-18 所示。

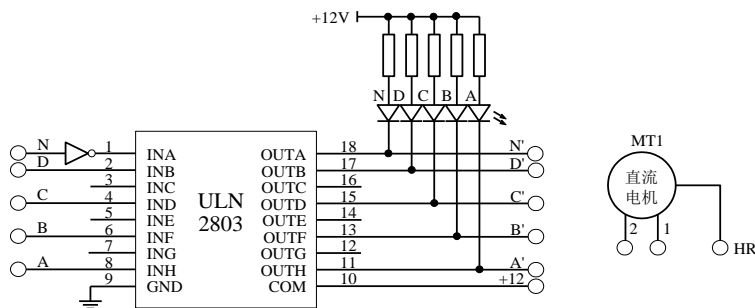


图 1-18 驱动电路和直流电机单元

17. 温度控制单元

该单元由7805 芯片产生+5V 的稳定电压和一个24 欧的电阻构成回路。其线路图连接如图1-19 所示。

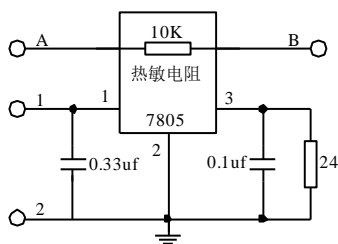


图 1-19 温度控制单元

18. 步进电机单元

该单元提供一个四项八拍的步进电机，如图1-20 所示。

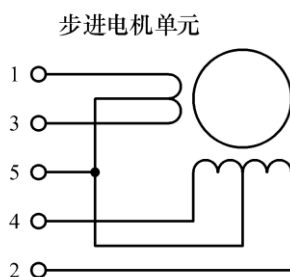


图 1-20 步进电机单元

19. 开关及LED 显示单元

该单元包括十六组拨动开关及LED 显示灯，用于输出和指示逻辑电平(正逻辑)。当显示灯亮时表示逻辑高电平，灭时表示逻辑低电平。其电路连接如图1-21 所示。

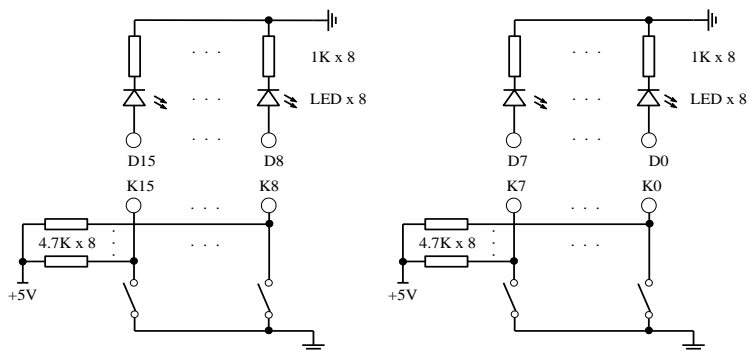


图 1-21 开关及 LED 显示单元

20. 时钟源单元

该单元提供一个1.8432MHz 的晶振电路，主要作为16550 时钟输入。另外还有两个十分频电路，将1.8432MHz 分频得到184.32KHz 和18.432KHz，如图1-22 所示。

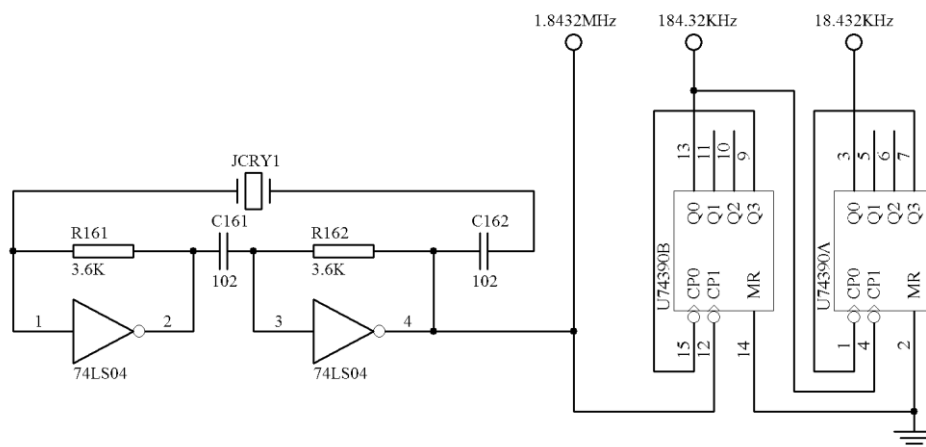


图1-22 时钟源单元

21. PWM 转换电路

该单元提供了一个PWM 脉冲产生电路，在IN 端输入0~5V 电压，OUT 输出脉冲的占空比会跟随产生相应的变化。其线路如图1-23 所示。

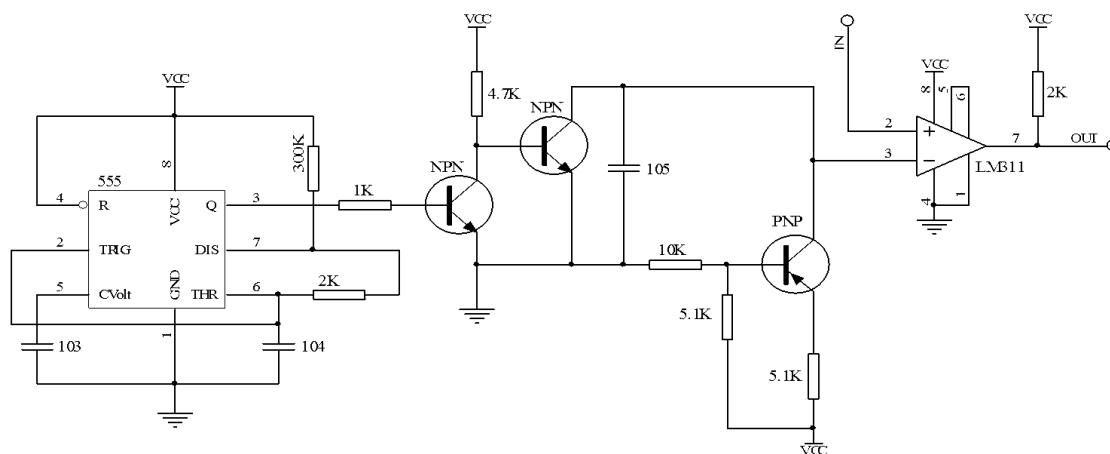


图1-23 PWM 转换电路

22. 单脉冲单元

该单元包括两个单脉冲触发器，由74LS00芯片和微动开关等构成两路R-S 触发器。单脉冲输出分上沿和下沿，分别以“+”和“-”表示。其线路如图1-24 所示。

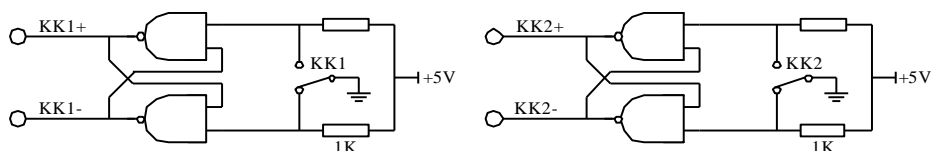


图 1-24 单脉冲单元

第二章 32 位微机原理软件实验

实验一 四则运算

一. 实验目的

- 1) 熟悉汇编语言程序的框架结构，掌握顺序结构的编程方法。
- 2) 熟悉 Tddebug 调试环境和 Turbo Debugger 的使用。
- 3) 理解 X86 内存数据的组织方式。
- 4) 理解基本的内存寻址方式。

二. 实验任务

完成 32 位无符号数的加法、减法，16 位乘以 16 位，32 位除以 16 位除法的四则运算练习。

三. 原理提要

- 1) 字节序

字节序分为大端模式 (Big-Endian) 和小端模式 (Little-Endian)，定义如下：

- 1) Little-Endian 就是低位字节排放在内存的低地址端，高位字节排放在内存的高地址端。
- 2) Big-Endian 就是高位字节排放在内存的低地址端，低位字节排放在内存的高地址端。

例如：32bit 宽的数 0x12345678 在小端模式以及大端模式 CPU 内存中的存放方式（假设从地址 0x4000 开始存放）为：

内存地址	小端模式	大端模式
0x4000	0x78	0x12
0x4001	0x56	0x34
0x4002	0x34	0x56
0x4003	0x12	0x78

★ Tips

常见 CPU 的字节序：

Big Endian : PowerPC、IBM、Sun

Little Endian : x86、DEC

ARM 既可以工作在大端模式，也可以工作在小端模式。

常见文件的字节序：

Adobe PS - Big Endian

BMP - Little Endian

GIF - Little Endian

JPEG - Big Endian

RTF - Little Endian

四. 预习要求

阅读参考代码，理解逐条语句的意义和实验的基本内容，并回答以下问题：

- 1) 理解 X86 内存数据的组织方式：小端模式。分别写出加、减、乘、除 4 个运算的运算数，及其在内存中的存放方式；
- 2) 根据参考代码中的加法运算，分别补全减法、乘法、除法的程序代码，注意运算数及运算结果的数据宽度。

五. 参考代码

```
DATA    SEGMENT
A       DW      1234H, 5678H           ;被加数
B       DW      0FEDCH, 123AH         ;加数
C       DW      2   DUP (?)           ;预留和
AD      DW      0FEDCH, 0BA98H        ;被减数
BD      DW      1234H, 789AH          ;减数
CD      DW      2   DUP (?)           ;预留差
A1      DW      0D678H                ;被乘数
B1      DW      0012H                 ;乘数
C1      DW      2   DUP (?)           ;预留积
A2      DW      5678H, 0234H          ;被除数
B2      DW      0F234H                ;除数
C2      DW      2   DUP (?)           ;预留商,余数
DATA    ENDS
```

```
STACK1  SEGMENT STACK
        DB      100 DUP(?)
STACK1  ENDS
```

```
CODE    SEGMENT
        ASSUME  CS:CODE,DS:DATA,SS:STACK1
```

```
START   PROC    FAR
        PUSH    DS                    }
        MOV     AX, 0                  } ;标准序
        PUSH    AX
        MOV     AX, DATA
        MOV     DS, AX

        MOV     AX, A                  }
        ADD     AX, B                  } ;32位加32位
        MOV     C, AX
        MOV     AX, A+2
        ADC     AX, B+2
        MOV     C+2, AX
```

在此补全代码… …

```
        RET
START   ENDP
```

```
CODE    ENDS
        END     START
```

六. 思考问题

- 1) 若需进行有符号数的运算, 需要注意什么问题? 如何实现?
- 2) 上述实验中, 我们在 80X86 的实模式中实现了 32 位的四则运算。例如, 乘法运算为 16 位乘以 16 位, 运算结果为 32 位。请思考如何利用 32 位指令, 实现 64 位的四则运算?

七. 实验报告内容

- 1) 实验目的和实验内容。
- 2) 写出自己补全的代码, 包含适当的注释以说明编写思路。
- 3) 记录数据段存放的原始数据, 和程序执行结束后的数据。

DS:0000								
DS:0008								
DS:0010								
DS:0018								
DS:0020								
DS:0028								

- 4) 根据以上数据段内容, 找出并记录和、差、积、商、余数运算结果。
- 5) 心得体会和对实验的意见及建议。

实验二 数据统计

一. 实验目的

- 1) 熟悉汇编语言程序的框架结构，掌握循环程序的设计方法。
- 2) 熟悉常用的条件跳转指令。
- 3) 熟悉有符号数的运算。

二. 实验任务

本实验要求通过求某数据区内负数的个数来表现循环程序的结构形式。要求实验程序在数据段中存放一组数据，分类统计数据中正数、负数和零的个数，并分别存入内存变量 Positive、Negative 和 Zero 中。将所有数据累加求和，存入 SUM 中。

三. 原理提要

- 1) 循环结构

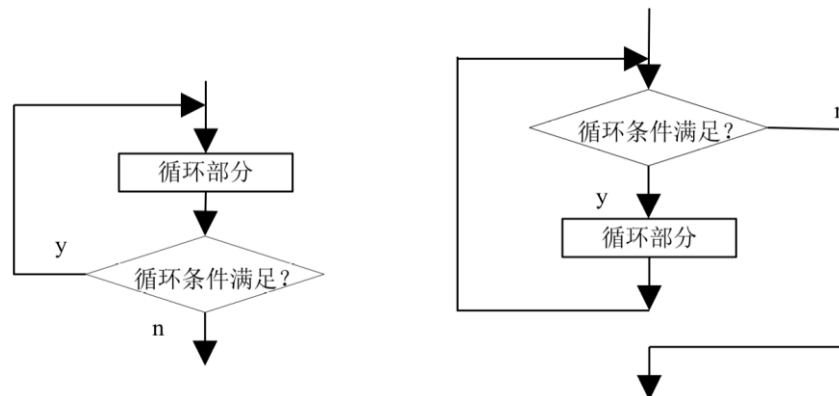


图 2-2-1 循环结构

常见的循环结构如图 2-2-1 所示，包括 Do-While 循环和 While 循环。如：循环次数是已知的，可用 LOOP 指令来构造循环；当循环次数是未知或不定的，则可用条件跳转指令来构成循环结构。

- 2) 有符号数和无符号数

计算机中的有符号数有三种表示方法，即原码、反码和补码。三种表示方法均有符号位和数值位两部分，规定高位为符号位，都是用 0 表示“非负”，用 1 表示“负”。

在计算机系统中，数值一律用补码来表示和存储。原因在于，使用补码，可以将符号位和数值域统一处理；同时，加法和减法也可以统一处理。也就是说，CPU 本身不识别操作数是否有符号，按照补码运算规则，有符号和无符号的数值结果都是一样的。

四. 预习要求

参考代码只给出了程序框架，根据自己的理解将缺失的代码补全，以达到实验任务中规定的要求。并考虑以下问题：

- 1) 能否从数据定义上区分其是否为有符号数？
- 2) 如何防止运算结果溢出？
- 3) 如何判断累加结果的符号？

五. 参考代码

```
DATA    SEGMENT
NUM     DB    12H,88H,82H,89H,33H,90H,0H,10H,0BDH,01H
Positive DB    DUP (?)
Negative DB    DUP (?)
Zero DB    DUP (?)
SUM DW 2 DUP (?)
DATA    ENDS
```

```
STACK1   SEGMENT STACK
          DB    100 DUP(?)
STACK1   ENDS
```

```
CODE     SEGMENT
          ASSUME CS:CODE,DS:DATA,SS:STACK1
```

```
START  PROC      FAR
        PUSH      DS
        MOV        AX,    0
        PUSH      AX
        MOV        AX,    DATA
        MOV        DS,    AX
```

} ;标准序

在此补全代码… …

```
        RET
START ENDP
CODE   ENDS
        END      START
```

六. 探究内容

- 1) 若需要将程序得到的结果显示在屏幕上，如何处理？
- 2) 利用某种排序算法，对原始数据进行排序。

七. 实验报告内容

- 1) 实验目的和实验内容。
- 2) 写出自己补全的代码，包含适当的注释以说明编写思路。
- 3) 记录数据段存放的原始数据,和程序执行结束后的数据。

DS:0000								
DS:0008								

- 4) 根据以上数据段内容，找出并记录数据中正数、负数和零的个数，以及求和结果。
- 5) 心得体会和对实验的意见及建议。

实验三 代码转换

一. 实验目的

- 1) 掌握 ASCII 码转换的基本方法。
- 2) 学会 INT21 功能调用, 掌握人机对话的设计方法。
- 3) 进一步熟悉 Tddebug 调试环境和 Turbo Debugger 的使用。

二. 实验任务

从键盘输入小写字母(最多 20 个), 以 “.” 号作为结束标志, 将其变换成相应的大写字母输出在屏幕上。

三. 实验分析

输入小写字母用 INT 21 的 0AH 号功能, 将读入的数据存放在缓冲区 SMALL 中, 其中 SMALL 的第一个字节指出缓冲区能容纳的字节数, 不能为 0(程序暂定为 50), 第二个字节保留, 以用作存放实际键入的输入字符的个数; 从第三个字节开始存放从键盘上输入字符的 ASCII 码, 所以转换时要从 SMALL 的第三个字节, 即 SMALL+2 开始。

SMALL 中存放的是小写字母的 ASSII 码, 将此值减去 20H, 即为大写字母的 ASSII 码, 将其依次存放在名为 CAPITAL 的数据段中, 然后用 INT 21 中的 09H 功能输出。 主程序流程图如图 2-3-1 所示。

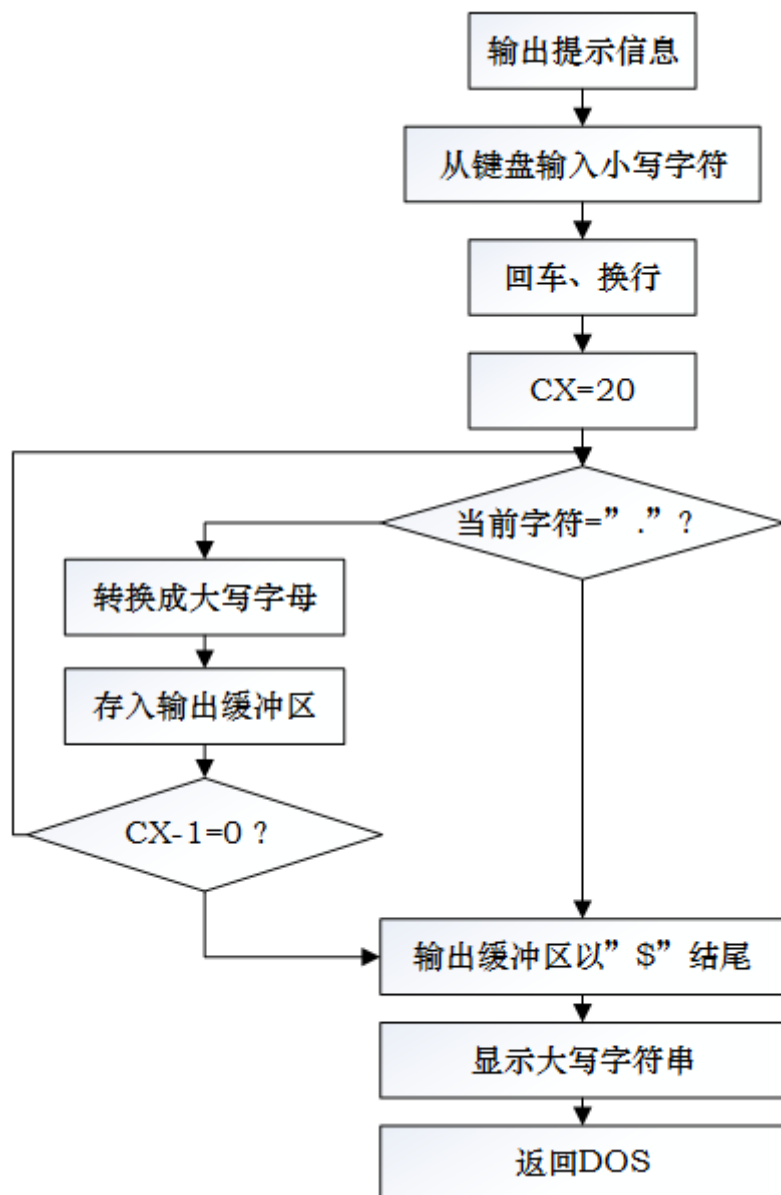


图 2-3-1 主程序流程图

四. 原理提要

1) 宏定义

若汇编程序中经常要用到一个程序段,可将其定义成一条宏指令,当汇编程序处理到宏指令时,会自动用宏体代换它而扩展成原来的程序段(宏扩展)。

宏指令由宏汇编程序识别,在程序汇编时完成宏扩展的处理,其目的仅为简化源程序的书写,缩短源程序的长度,并未缩短其目标代码的长度。

在本实验的参考代码中使用到宏定义,调试时请留意宏扩展。

2) INT 21H 系统功能调用

一般来说,有很多程序需要显示输出提示运行的状况和结果,有的还需要将数据区中的内容显

示在屏幕上。本实验要求将指定数据区的数据以十六进制数形式显示在屏幕上，并利用 DOS 功能调用完成一些提示信息的显示。通过本实验，初步掌握实验系统配套操作软件的使用。

实验中所使用 DOS 功能调用 (INT 21H) 说明如下：

(1) 显示单个字符输出

入口：AH=02H

调用参数：DL=输出字符

(2) 显示字符串

入口：AH=09H

调用参数：DS:DX=串地址，' \$ ' 为结束字符

(3) 键盘输入

入口：AH=0AH

调用参数：DS:DX=输入缓冲区地址，首字节为缓冲区字节长度，第二字节为实际输入的字符计数

(4) 返回 DOS 系统

入口：AH=4CH

调用参数：AL=返回码

五. 预习要求

参考代码只给出了程序框架，根据自己的理解将缺失的代码补全，以达到实验任务中规定的要求。

六. 参考代码

```
CRLF    MACRO
        MOV     DL,  0DH
        MOV     AH,  02H
        INT     21H
        MOV     DL,  0AH
        MOV     AH,  02H
        INT     21H
    ENDM
        } ;宏定义回车,换行

DATA    SEGMENT
MES1    DB      'PLEASE INPUT THE SMALL LETTER,ENDED WITH ".":$'
MES2    DB      'THE CAPTAL LETTER IS:$'
SMALL    DB      50                ; 预留键盘输入缓冲区长度为50个
        DB      ?                ; 预留实际键盘输入字符数的个数
```

```

        DB    50  DUP(?)
CAPITAL DB    50  DUP(?)      ; 预留大写字母缓冲区长度为50个
DATA    ENDS

```

```

STACK1 SEGMENT STACK
        DB    100  DUP (?)
STACK1 ENDS

```

```

CODE    SEGMENT
        ASSUME CS:CODE,DS:DATA,SS:STACK1
START PROC    FAR
        PUSH    DS
        MOV     AX, 0
        PUSH    AX
        MOV     AX, DATA
        MOV     DS, AX

        MOV     AH, 9      }
        LEA     DX, MES1   } ;输出提示信息MES1
        INT     21H
        CRLF                    ;宏调用

        MOV     AH, 0AH    }
        LEA     DX, SMALL1 } ; 接收小写字符串
        INT     21H
        CRLF                    ;宏调用

```

在此补全代码… …

```

KE:      MOV     AL, '$' }
        MOV     [DI], AL } ;大写字符串后加 “$”

        MOV     DX, OFFSET MES2 }
        MOV     AH, 9             } ; 输出提示信息MES2
        INT     21H
        CRLF                    ;宏调用

        MOV     DX, OFFSET CAPITAL }
        MOV     AH, 9             } ; 输出大写字符串
        INT     21H
        RET

START    ENDP
CODE     ENDS
END      START

```

七. 探究内容

- 1) 若从键盘输入的字符非小写字母，其 ASCII 码减去 20H 后输出为无关字符，如何在程序中避免以上问题？

八. 实验报告内容

- 1) 实验目的和实验内容。
- 2) 写出自己补全的代码，包含适当的注释以说明编写思路。
- 3) 心得体会和对实验的意见及建议。

实验四 数据块移动

一. 实验目的

- 1) 进一步掌握主程序、子程序设计方法。
- 2) 掌握人机对话的设计方法。
- 3) 进一步熟悉 Tddebug 调试环境和 Turbo Debugger 的使用。

二. 实验任务

本实验要求将指定数据区的数据搬移到另一块内存空间中，并通过子程序调用的方法将搬移的数据显示在屏幕上。

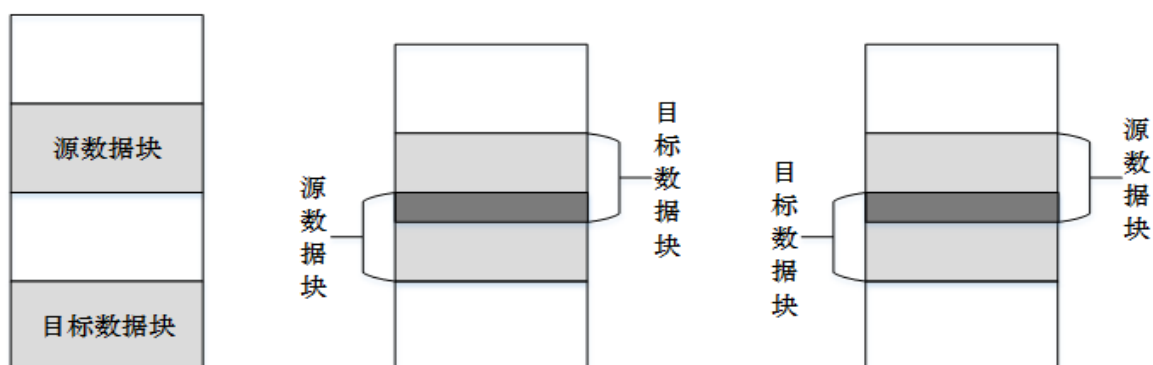


图 2-4-1 源数据块和目标数据块在存储器中的位置示意

源数据块和目标数据块在存储中的位置可能有三种情况，如图 2-4-1 所示。对于两个数据块分离的情况，数据的传送从数据块的首地址开始，或者从数据块的末地址开始均可。但对于有部分重叠的情况，则要加以分析，否则重叠部分会因搬移而遭到破坏。

所以搬移过程可以通过以下两个方式完成：

当源数据块首地址>目标块首址时，从数据块的首地址开始传送数据；

当源数据块首地址<目标块首址时，从数据块的末地址开始传送数据。

三. 原理提要

在汇编程序设计中，用户通常会将常用的具有特定功能的程序段编制成子程序使用。一般过程定义伪操作的格式如下：

```
procedure name PROC Attribute
.....
procedure name ENDP
```

其中 Attribute 是指类型属性，可以是 NEAR 或 FAR，调用程序和过程在同一个代码段中使用 NEAR 属性，不在同一个代码段中，使用 FAR。

四. 预习要求

参考代码只给出了程序框架，根据自己的理解将缺失的代码补全，以达到实验任务中规定的要求。并考虑以下问题：

- 1) 参考代码仅假设为上图中的情况 1，自行模拟图 2-4-1 中数据块的另外两种情况，构造相应的代码完成数据块的移动；
- 2) 通过子程序调用的方法将搬移的数据显示在屏幕上。注意：数据输出至屏幕前，须将其转换成对应的 ASCII 码。

五. 参考代码

```
STACK1    SEGMENT STACK
           DW 256 DUP(?)
STACK1    ENDS
DATA      SEGMENT
MES1      DB  'The data in buf2 are:',0AH,0DH,'$'
BUF1      DB
11H,22H,33H,44H,55H,66H,77H,88H,99H,0AAH,0BBH,0CCH,0DDH,0EEH,0FFH,00H
BUF2      DB 20H    DUP(0)
DATA      ENDS
CODE      SEGMENT
           ASSUME CS:CODE,DS:DATA
START:    MOV  AX,DATA
           MOV  DS,AX
           MOV  SI,OFFSET BUF1
           MOV  DI,OFFSET BUF2
           

在此补全代码… …


           RET
CODE ENDS
           END START
```

六. 探究内容

- 1) 利用某种排序算法，对原始数据进行排序后输出至屏幕。

七. 实验报告内容

- 1) 实验目的和实验内容。
- 2) 写出自己补全的代码，包含适当的注释以说明编写思路。
- 3) 心得体会和对实验的意见及建议。

实验五 描述符和描述符表实验（选做）

一. 实验目的

- 1) 熟悉保护模式的编程格式。
- 2) 掌握描述符的声明方法。
- 3) 掌握使用选择符访问段的寻址方法。

二. 实验任务

实现在一个 0 级代码段中将源数据段中的一段数据复制到目标数据段中。其中所有段的段描述符均放置在 GDT 中。

1. 实验分析

为了实现在 0 级代码段中完成数据传输，实验程序中需要安排一个 0 级代码段和两个 0 级数据段（可以是 0~3 级任一级别的数据段）。

在程序开始声明一个数据段‘DSEG’，来描述这三个段的描述符，其中有代码段描述符 Scode，源数据段描述符 DataS 和目标数据段描述符 DataD，将它们相应的选择子分别定义为 Scode_Sel，DataS_Sel，DataD_Sel。按照实验程序编写格式的约定及描述符的格式定义，为这三个段分别定义描述符：

(1) 代码段描述符：Scode Desc <Clen,CSEG,,ATCE,,>;

段属性说明：

G :	0	; 以字节为段界限粒度
D :	0	;是 16 位的段
P :	1	;描述符对地址转换有效/该描述符对应的段存在
DPL :	0	;0 级段
DT :	1	;描述符描述的是存储段
TYPE:	0 x 8	;只执行段

段基地址说明：定义代码段的标号为 CSEG，则在段基地址处填写 CSEG，为调试器提供重定位信息。

段界限说明：段界限定义为 Clen。

(2) 源数据段描述符：DataS Desc <DLEN,DSEG1,,ATDW,,>;

段属性说明：

G :	0	; 以字节为段界限粒度
-----	---	-------------

D : 0 ; 是 16 位的段
P : 1 ; 描述符对地址转换有效/该描述符对应的段存在
DPL : 0 ; 0 级段
DT : 1 ; 描述符描述的是存储段
TYPE: 0 x 2 ; 可读写段

段基地址说明：定义源数据段标号为 DSEG1，则在段基地址处填写 CSEG，为调试器提供重定位信息。

段界限说明：定义段界限为 DLEN。

(3) 目标数据段描述符：Datad Desc <BUFLEN,DSEG2,,ATDW,,>

目标数据段描述符的内容基本与源数据段的内容相同，只要修改段基地址和段界限的定义即可。

为了给装入程序提供重定位信息，三个存储段描述符中地址的低 16 位，用每个描述符对应段的标号来填写。在程序装入内存时，调试系统会根据地址的低 16 位定位该段对应的真实物理地址，并将该地址写入描述符中（系统没有使用分页机制，线性地址等价于物理地址）。在实验中可查询 GDT 表来确定每个段的真实物理地址。

在程序定义过程中，首先使用一个全“F”的描述符作为定义的开始，然后定义代码段描述符 Scode、源数据段的描述符 DSEG1 和目标数据段描述符 DSEG2。为了区分 LDT 表和 GDT 表的定义，再使用一个全“F”的描述符作为界限。由于本实验中不使用 LDT 表，则再使用一个全“F”的描述符结束描述符的声明。

本程序可实现将一个数据段中数据搬运到另一个数据段的过程。传输过程中可使用 DS，ES 两个段寄存器，其中 DS 装入源数据段的选择符 DataS_Sel，ES 装入目标数据段的选择符 DataD_Sel。在实验程序的最后使用“INT 0FFH”指令，正常结束程序运行。

三. 实验步骤

- (1) 进入纯 DOS 环境，运行 Tddebug 集成操作软件。
- (2) 运行 Tddebug 软件，使用 Alt+E 选择 Edit 菜单项进入程序编辑环境。按实验要求编写程序。实验参考流程图如图 2-5-1。
- (3) 程序编写完后保存退出，使用 Alt+C 选择 Compile 菜单中的 Compile 和 Link 命令对实验程序进行编译、连接。
- (4) 编译输出信息表示无误后，使用 Alt+P 选择 Pmrun 命令装入实验程序，如果装入成功，屏幕上会显示“Load OK!”，否则，会给出相应的错误提示信息。
- (5) 若程序装入成功，则进入保护模式调试环境。在命令输入行使用 GDT 命令查询系统的 GDT

表, 并且查看实验程序中声明的代码段、数据段描述符在 GDT 表中的位置以及对应段的物理地址、段属性、段界限等。

(6) 使用 F7 单步执行程序, 执行 LLDT AX 语句后, 使用 LDT 命令查看 LDT 局部描述符表的内容。

(7) 根据 LDT 内容使用 D 命令查看源数据区的数据和目的数据区数据内容。

(8) 按 F9 运行程序, 如果程序正常运行结束, 命令显示区中将显示 “Correct Running”。

(9) 运行结束后, 再次查看目的数据区内容, 观察数据传输是否正确。

四. 程序流程图

全局描述符表实验参考流程图见图 2-5-1

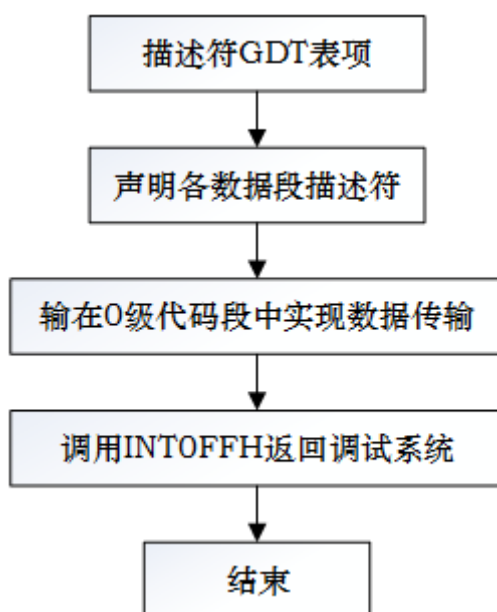


图 2-5-1 全局描述符表实验参考流程图

五. 参考代码

```
.386P
DESC    STRUC        ; 定义描述符结构
        LIMITL    DW 0
        BASEL     DW 0
        BASEM     DB 0
        ATTR      DB 0
        LIMITH    DB 0
        BASEH     DB 0
DESC    ENDS

        ATCE = 98H      ; 定义特殊常量
        ATDR = 90H
        ATDW = 92H
```



```

DSEG SEGMENT USE16      ;声明开始初始化GDT表中的描述符
    GDT LABEL BYTE
    ID1 DESC <0FFFFH,0FFFFH,0FFH,0FFH,0FFH,0FFH> ;标记1
    ID2 DESC <0FFFFH,0FFFFH,0FFH,0FFH,0FFH,0FFH> ;标记2
    ID3 DESC <0FFFFH,0FFFFH,0FFH,0FFH,0FFH,0FFH> ;标记3
    SCODE DESC <0FFFFH,CSEG,,ATCE,,>           ;代码段描述符
    DATAS DESC <D1LEN,DSEG1,,ATDW,,>           ;源数据段描述符
    DATAD DESC <BUFLen-1,DSEG2,,ATDW,,> ;目标数据段描述符
    GDTLEN = $-GDT                ;目标数据段描述符
    SCODE_SEL = SCODE-GDT          ;代码段选择子
    DATAS_SEL = DATAS-GDT          ;源数据段选择子
    DATAD_SEL = DATAD-GDT          ;目标数据段选择子
DSEG ENDS

DSEG1 SEGMENT           USE16      ;定义源数据段
    BUF DB 00H,11H, 22H, 33H, 44H, 55H, 66H, 77H
        DB 88H,99H,0AAH,0BBH,0CCH,0DDH,0EEH,0FFH
        DB 240 DUP(0)
    D1LEN = $-1
DSEG1 ENDS

DSEG2 SEGMENT USE16      ;定义目标数据段
    BUFLen = 256
    BUFFER DB BUFLen DUP(0)
DSEG2 ENDS

CSEG SEGMENT USE16
ASSUME CS:CSEG, DS:DSEG
START PROC
    MOV AX, DATAS_SEL      ;装入源数据段选择子
    MOV DS, AX
    MOV AX, DATAD_SEL      ;装入目标数据段选择子
    MOV ES, AX
    CLD
    XOR SI, SI
    XOR DI, DI
    MOV CX, 32
M1:  MOVSB
    LOOP M1
    INT 0FFH                ; 返回调试系统
START ENDP
CLEN = $-1
CSEG ENDS
END START

```

实验六 局部描述符表实验（选做）

一. 实验目的:

- 1) 熟悉编程格式，掌握通过 ldt 表访问段的编程方法。

本实验与上一实验所完成的功能相同，但要求将代码段安排在全局描述符表中，而将数据段安排在局部描述符表中。

二. 实验分析

本实验需要为代码段和数据段分别声明描述符，由于要求将数据段的描述符放入 LDT 表中，所以实验程序需要建立一张局部描述符表，并在 GDT 表中声明 LDT 表对应的描述符。描述符声明完成，还需要为它们定义相应的选择子。

实验程序在 DSEG 段中描述 GDT 表中的描述符。先用一个全“F”的描述符作为定义的开始，然后定义主程序段和 LDT 表描述符，然后使用一个全“F”的描述符作为区分于 LDT 表的界限。在 DSEG 段后，用 DSEG1 段来描述 LDT 表中的描述符，其中包括源数据段描述符目标数据段描述符。在 DSEG1 段的末尾再使用一个全“F”描述符作为描述符声明的结尾。

由于主代码段需要访问的段是在 LDT 表中声明的，所以在程序的初始需要执行装载 LDTR 的指令。装载 LDT 表使用的指令如下：

```
MOV    AX, LDT_Sel
LLDT   AX
```

(1) LDT 表对应段描述符: LDTable Desc <LDTLen-1,DSEG1,,ATLDT,,>

段属性说明:

```
G : 0      ; 以字节为段界限粒度
D : 0      ; 是 16 位的段
P : 1      ; 描述符对地址转换有效/该描述符对应的段存在
DPL : 0    ; 0 级段
DT : 0     ; 描述符描述的是系统段或门描述符
TYPE: 0x8  ; LDT 表
```

段基地址说明: 需要在重定位后确定，但可以知道，该描述符对应的数据段是 DSEG1

段界限说明: 段界限为 LDTLen-1

ATLDT EQU 82h ;局部描述符表段类型值

(2) 数据段选择子

实验中的两个数据段均在 LDT 表中声明，则描述符对应的段选择符应该标记出来

```
TIL EQU 04h
```

```
DataS_Sel = DataS-LDT+TIL
```

```
DataD_Sel = DataD-LDT+TIL
```

三. 实验步骤

(1) 进入纯DOS环境，运行Tddebug集成操作软件。

(2) 运行 Tddebug 软件，使用 Alt+E 选择 Edit 菜单项进入程序编辑环境。按实验要求编写程序。

实验参考流程图如图 2-6-1 所示。

(3) 程序编写完后保存退出，使用Alt+C选择Compile菜单中的Compile和Link命令对实验程序进行编译、连接。

(4) 编译输出信息表示无误后，使用Alt+P选择Pmrun命令装入实验程序，如果装入成功，屏幕上会显示“Load OK!”，否则，会给出相应的错误提示信息。

(5) 若程序装入成功，则进入保护模式调试环境。在命令输入行使用GDT命令查询系统的GDT表，并且查看实验程序中声明的代码段、数据段描述符在GDT表中的位置以及对应段的物理地址、段属性、段界限等。

(6) 使用F7单步执行程序，执行LLDT AX语句后，使用LDT命令查看LDT局部描述符表的内容。

(7) 根据LDT内容使用D命令查看源数据区的数据和目的数据区数据内容。

(8) 按F9运行程序，如果程序正常运行结束，命令显示区中将显示“Correct Running”。

(9) 运行结束后，再次查看目的数据区内容，观察数据传输是否正确。

四. 程序流程图

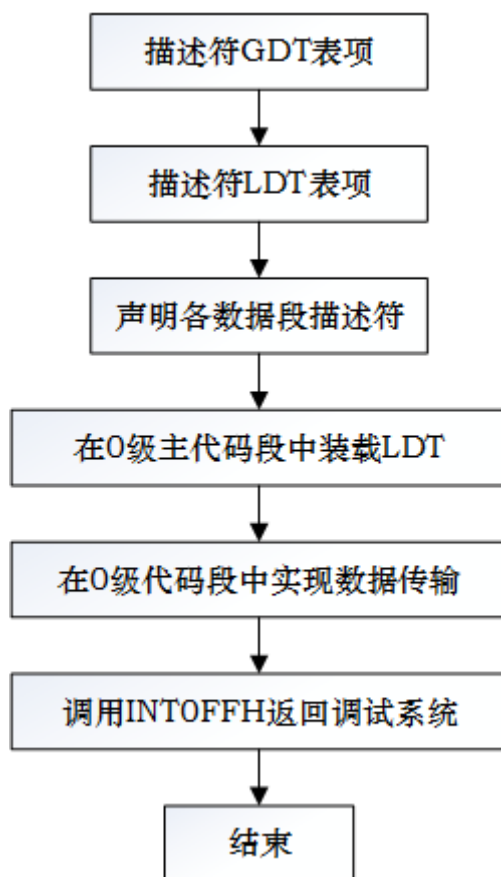


图 2-6-1 局部描述符表实验参考流程图

五. 参考实验程序

;实验结果： 可以用dump命令检查数据区，验证程序的运行结果

;编程注意事项： ldt表所在段对应的描述符必须在gdt表中声明，程序起始代码段对应描述符也应该在gdt表中声明

```

INCLUDE 386SCD.INC
DSEG SEGMENT use16
GDT LABEL BYTE ;全局描述符表
ID1 Desc <0ffffh,0ffffh,0ffh,0ffh,0ffh,0ffh> ;空描述符
Mcode Desc <0ffffh,CSEG,,ATCE,,> ;代码段描述符
LDTable Desc <LDTLen-1,DSEG1,,ATLDT,,> ;局部描述符表段的描述符
GDTLen = $-GDT ;全局描述符表长度
MCode_Sel = MCode-GDT
LDT_Sel = LDTable-GDT ;局部描述符表段的选择子
ID2 Desc <0ffffh,0ffffh,0ffh,0ffh,0ffh,0ffh>
DSEG ENDS ;数据段定义结束

DSEG1 SEGMENT use16
LDT LABEL BYTE ;局部描述符表
DataS Desc <0ffffh,0,11h,ATDW,,> ;源数据段描述符
DataD Desc <0ffffh,Dseg2,,ATDW,,> ;目标数据段描述符
DataS_Sel = DataS-LDT+TIL
DataD_Sel = DataD-LDT+TIL
  
```

```

LDTLen    =        $-LDT                ;局部描述符表长度
ID3       Desc    <0ffffh,0ffffh,0ffh,0ffh,0ffh,0ffh>
DSEG1     ENDS
Dseg2     Segment use16
BufLen    =        256                  ;缓冲区字节长度
Buffer     DB      BufLen DUP(0)        ;缓冲区
Dseg2     ends
CSEG      SEGMENT use16                  ;16位代码段
        ASSUME CS:CSEG
Start     PROC
        mov     ax,LDT_Sel
        ldt ax
        mov     ax, DataS_Sel
        mov     ds,ax                    ;加载源数据段描述符
        mov     ax, DataD_Sel
        mov     es,ax                    ;加载目标数据段描述符
        cld
        xor     si,si
        xor     di,di                    ;设置指针初值
        mov     cx,BufLen/4              ;设置4字节为单位的缓冲区长度
        repz    movsd                     ;传送
        int 0ffh
Start     ENDP
CSEG     ENDS                            ;代码段定义结束
        END      Start

```

第三章 32 位微机接口硬件实验

实验一 地址译码电路与 I/O 接口

一. 实验目的

- 1) 学习 3-8 译码器在接口电路中的应用。
- 2) 掌握地址译码电路的一般设计方法。
- 3) 理解输入输出接口的基本原理。

二. 实验任务

用 74LS138 译码器设计地址译码电路, 其 Y7 作为基本输入/输出单元的片选信号。参考实验电路如图 3-1-4 所示, 功能及流程如下:

- 1) 读入 74LS245 输入的八位数据, 在 74LS574 上输出, 用八位 LED 显示开关状态。
- 2) 当有键按下, 且读入的数字量为 1, 则八位 LED 从右向左依次循点亮, 否则重读数字量。
- 3) 再有键按下, 且读入的数字量为 2, 则八位 LED 交替亮, 否则重读数字量。
- 4) 再有键按下返回。

三. 原理提要

实验箱上有关单元的具体细节, 请查阅第一章的相关内容。

在微机接口电路中, 常采用 74LS138 译码器的输出用作为 I/O 端口或存储器地址片选信号。74LS138 有 3 个输入端 G1、G2A、G2B; 3 个控制端 A、B、C 及 8 个输出引脚, 其管脚如图 3-1-1 所示。当 3 个控制信号有效时, 来确定那个输出端为低电平, 而其它输出端为高电平, 该低电平信号即可作为片选信号。

3 个控制端 A、B、C 由地址线接入, 改变地址即可获得相应的片选信号。以 A、B、C 分别接地址线 A3、A4、A5 为例 (实验中可以改变控制地址线), I/O 地址见表 3-1-1:

输入接口一般用三态缓冲器, 本实验中 74LS245 三态缓冲器外接八位开关, 其开关状态作为数据传送给微机系统。74LS245 是 8 通道双向的三态缓冲器, 其管脚结构如图 3-1-2 所示。DIR 引脚控制缓冲器数据方向, DIR 为 0 表示数据由 A[7:0]至 B[7:0], DIR 为 1 表示数据由 B[7:0]至 A[7:0]。E 引脚为缓冲器的片选信号, 低电平有效。

输出接口一般用锁存器, 本实验中 74LS574 输入端连接总线的的数据, 输出端连接八位 LED。74LS574 管脚结构如图 3-1-3 所示。D[7:0]为输入数据线, Q[7:0]为输出数据线。CLK 引脚为锁存控制信号, 上升沿有效, 输入的数据锁存后送八位 LED。OC 引脚为锁存器的片选信号, 低电平有

效。

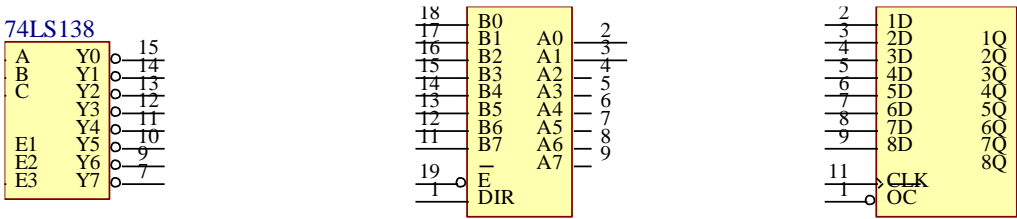


图 3-1-1 74LS138 译码器引脚 图 3-1-2 74LS245 缓冲器引脚 图 3-1-3 74LS574 锁存器引脚

表 3-1-1 地址译码电路输入及输出

地址 A7~A0	地址（16 进制）	C B A	输出片选
00000000	00H	000	Y0
00001000	08H	001	Y1
00010000	10H	010	Y2
00011000	18H	011	Y3
00100000	20H	100	Y4
00101000	28H	101	Y5
00110000	30H	110	Y6
00111000	38H	111	Y7

地址译码电路与 I/O 接口的实验电原理框图如图 3-1-4 所示。

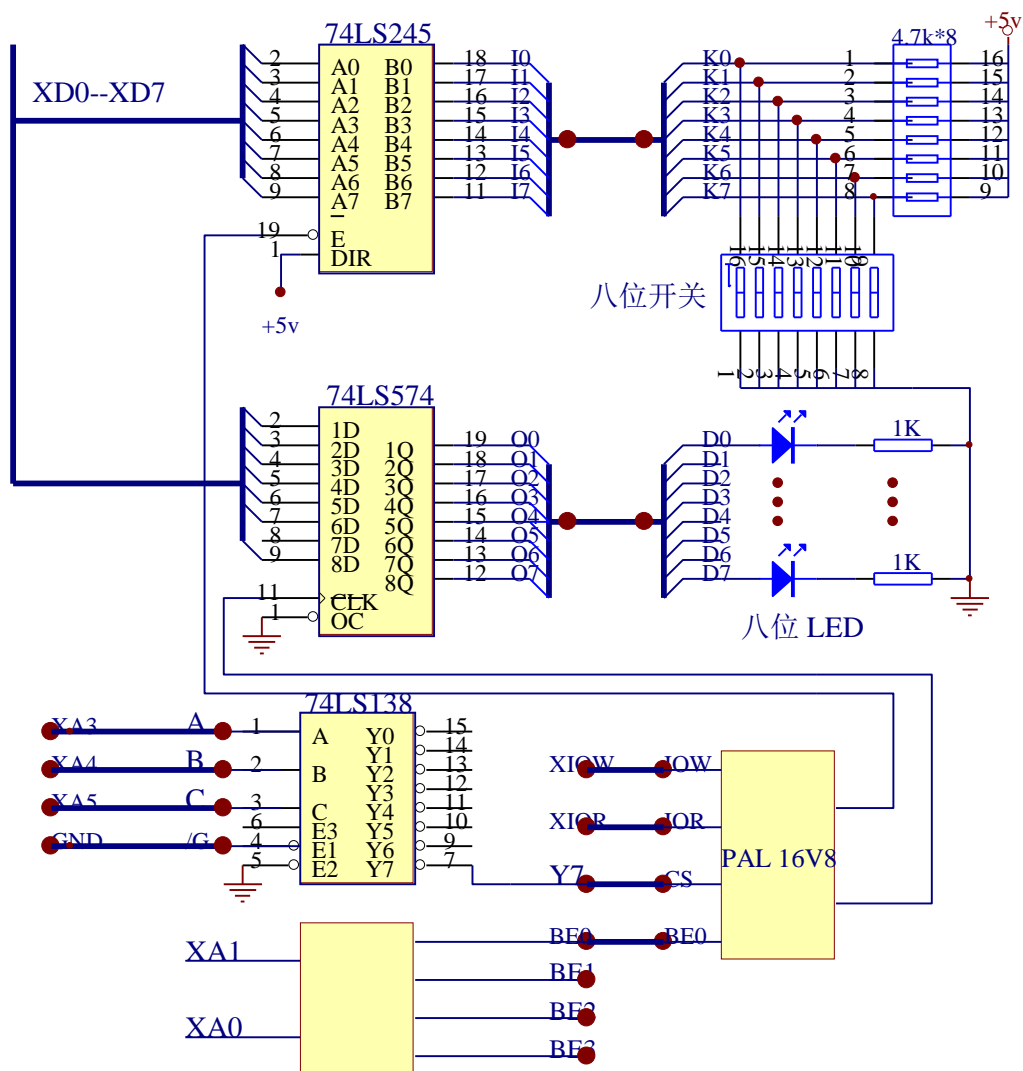


图 3-1-4 地址译码与 I/O 接口实验电原理框图

四. 预习要求

- 1) 熟悉实验软硬件原理，对照程序流程图仔细阅读参考代码。
- 2) 尝试加入自己的想法，改变参考代码的输出功能或控制流程。

地址译码电路与 I/O 接口设计流程图如图 3-1-5 所示。

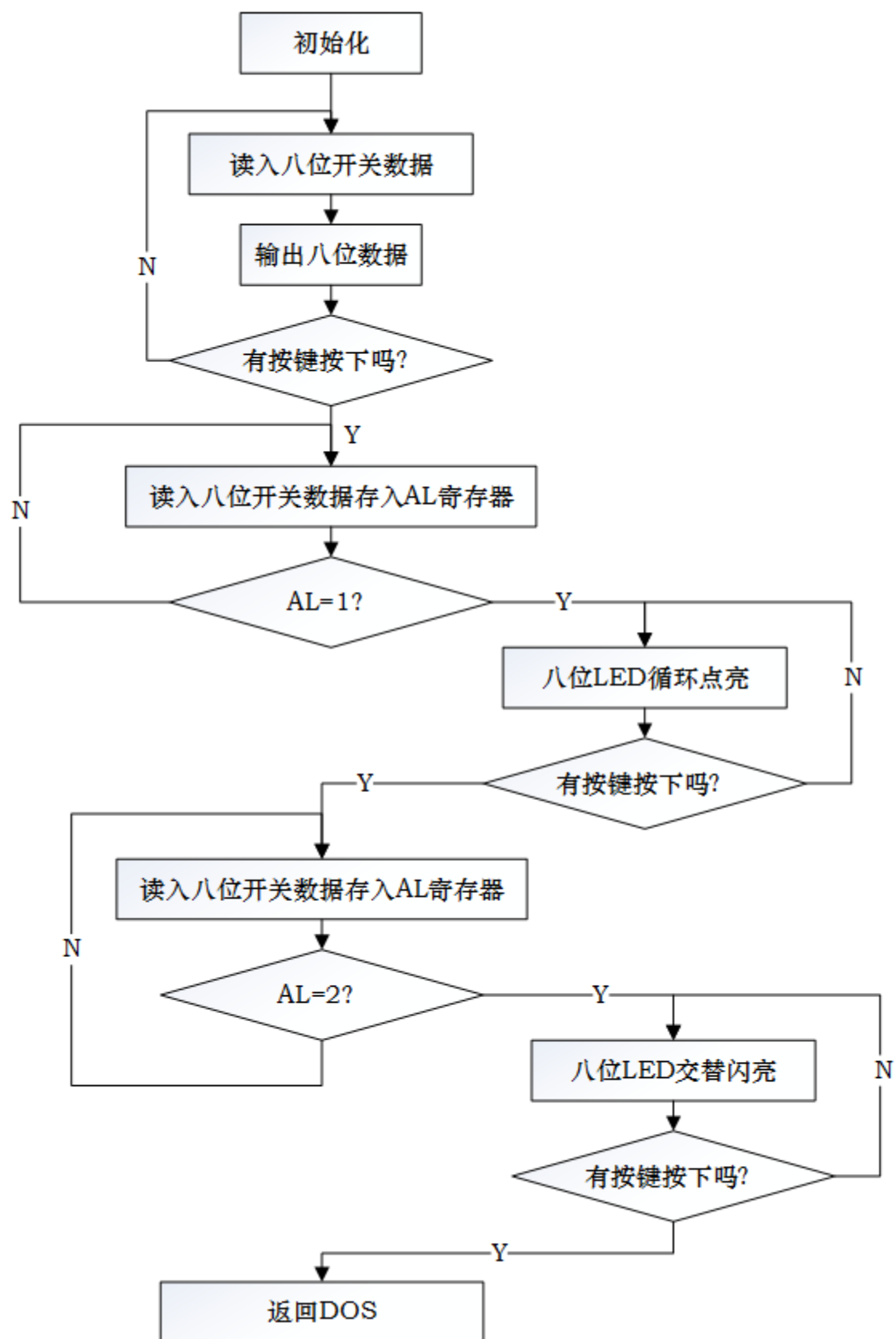



图 3-1-5 地址译码电路与 I/O 接口设计流程图

五. 单元自检

为保证实验单元工作正常,请在实验开始前进行快速单元自检,通过简单的 Debug 命令检查单元硬件是否可以正常工作。参考步骤如下:

1) 按照原理图完成连线。

2) 点击工具栏图标 

3) 进入 Debug 环境,通过键盘键入 “Debug” 命令:

```
C:\TangDu\PitPP\Asm>DEBUG
```

```
I 3038      ;读入当前八位开关状态, 改变八位开关状态,开关量亦不同
```

```
O 3038 FF   ;八位 LED 全亮
```

```
Q          ;退出自检
```

4) 返回 WINDOWS C:\TangDu\PitPP\Asm>EXIT

六. 参考代码

```
IOY0 EQU 3000H      ;片选IOY0对应的端口始地址
Y7 EQU IOY0+38H     ;译码电路输出Y7对应的端口地址
DATA SEGMENT
NUM DB 01H
DATA ENDS
STACK1 SEGMENT STACK
DW 256 DUP(?)
STACK1 ENDS
CODE SEGMENT
ASSUME CS:CODE,DS:DATA,SS:STACK1
START: MOV AX, DATA
MOV DS, AX
MOV DX, Y7          } ;读入开关量
IN AL, DX
OUT DX, AL
MOV DL, 0FFH
MOV AH, 6            } ;判断是否有按键按下
INT 21H
JZ START             ;无按键继续循环, 有则退出
L1: MOV DX, Y7
IN AL, DX            } ;读入开关量, 判断是否为1
CMP AL, 1
JNE L1
L2: MOV DX, Y7
MOV AL, NUM
OUT DX, AL           } ;八位LED从右向左依次循点亮
```

```

        ROL    AL,1
        MOV    NUM,AL
        CALL   DELAY

        MOV    DL,0FFH
        MOV    AH,6
        INT    21H
        JZ     L2
        JZ     L2           ;无按键继续循环，有则退出
L3:      MOV    DX,Y7
        IN     AL,DX
        CMP    AL,2
        JNE    L3           ; 读入开关量，判断是否为2

        MOV    NUM,55H
        MOV    DX,Y7
        MOV    AL,NUM
        OUT    DX,AL
        NOT    AL
        MOV    NUM,AL
        CALL   DELAY
        MOV    DL,0FFH
        MOV    AH,6
        INT    21H
        JZ     L4
        JZ     L4           ;无按键继续循环，有则退出

        MOV    AX,4C00H
        INT    21H           ;结束程序退出

DELAY PROC
        MOV    BX,8FFH
DELAY1: MOV    CX,0FFFFH
        LOOP   $
        DEC    BX
        JNZ    DELAY1
        RET
DELAY ENDP
CODE ENDS
END START

```

七. 探究内容

- 1) 上述实验方案中只使用了输入输出接口的低八位，请将上述实验方案改为 16 位输入输出，并当堂完成。

八. 实验报告

- 1) 实验目的和实验内容。
- 2) 通过简要的电原理框图和程序流程图，描述自己设计的实验方案。
- 3) 实验结果。
- 4) 心得体会和建议。

实验二 8254 定时/计数器

一. 实验目的

- 1) 掌握 8254 定时/计数器的各种工作方式及编程方法。

二. 实验任务

按照图 3-2-2 的要求连线，分别对 8254 芯片的 3 个定时/计数器编程，并选择合适的工作方式和初值，以达到如下的效果：

- 1) 定时/计数器 0，计数脉冲频率为 18.432KHz，OUT0 分两路输出，一路外接 2 位 LED，使其以亮 0.5 秒灭 0.5 秒循环闪亮，另一路作为计数器 1 的计数脉冲 CLK1。
- 2) 定时/计数器 1，OUT1 的输出外接 2 位 LED，使其以亮 3 秒灭 1 秒循环闪亮。
- 3) 定时/计数器 2 的计数脉冲来自单次脉冲单元，按压开关产生的脉冲作为计数器 2 的计数脉冲。OUT2 外接 2 位 LED，当按压开关到 17 次时 LED 长亮，并将按压开关的剩余次数将在屏幕上显示。

三. 原理提要

8254 是 Intel 公司生产的可编程/定时器。是 8253 的改进型，比 8253 具有更优良的性能。8254 具有以下基本功能：

- 1) 有 3 个独立的 16 位计数器；
- 2) 每个计数器可按二进制或十进制（BCD）计数；
- 3) 每个计数器可编程工作于 6 种不同工作方式；
- 4) 8254 每个计数器允许的最高计数频率为 10MHz（8253 为 2MHz）；
- 5) 8254 有读回命令（8253 没有），除了可以读出当前计数单元的内容外，还可以读出状态寄存器的内容。
- 6) 计数脉冲可以有规律的时钟信号，也可以是随机信号。计数初值公式为：

$$n = F_{clk} \div F_{out}$$

其中 F_{clk} 是输入计数脉冲的频率， F_{out} 是输出信号频率。

图 3-2-1 是 8254 的内部结构框图和引脚图，它主要由数据总线缓冲器、R/W 逻辑电路、控制寄存器和三个计数器组成。CLK 为计数脉冲输入端，GATE 为门控输入端，OUT 为计数器输出端。

8254 有六种工作方式，分别如下：

- 1) 方式 0：只计数一遍，计数到 0，OUT 为高电平，此正跃变信号可作为中断请求信号。不能自动恢复初值只有在写入另一初值时，开始新的计数。

- 2) 方式 1：可重触发的单拍脉方式。
- 3) 方式 2：频率发生器方式。
- 4) 方式 3：方波发生器。
- 5) 方式 4：软件触发选通方式。
- 6) 方式 5：硬件触发选通方式。

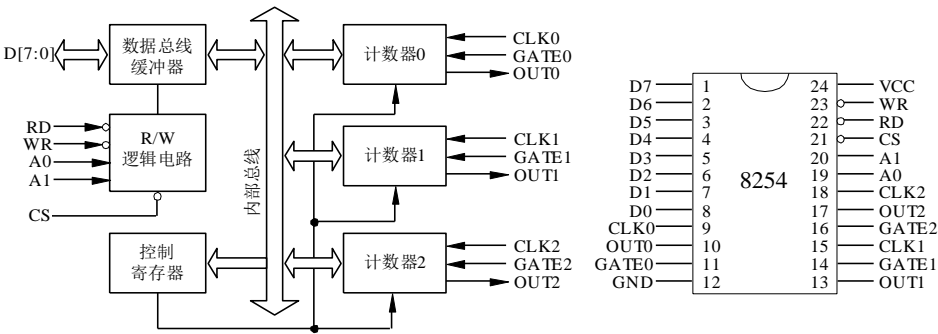


图 3-2-1 8254 的内部接口和引脚

8254 的控制字有两个：一个用来设置计数器的工作方式，称为方式控制字；另一个用来设置读回命令，称为读回控制字。这两个控制字共用一个地址，由标识位来区分。控制字格式如表 3-2-1 所示。读回控制字格式如表 3-2-2 所示。当读回控制字的 D4 位为 0 时，由该读回控制字 D1~D2 位指定的计数器的状态寄存器内容将被锁存到状态寄存器中。状态字格式如表 3-2-3 所示。

表 3-2-1 8254 的方式控制字格式

D7	D6	D5	D4	D3	D2	D1	D0
计数器选择		读/写格式选择		工作方式选择			计数码制选择
00—计数器 0		00—锁存计数值		000—方式 0			0—二进制数
01—计数器 1		01—读/写低 8 位		001—方式 1			1—十进制数
10—计数器 2		10—读/写高 8 位		010—方式 2			
11—读出控制 字标志		11—先读/写低 8 位 再读/写高 8 位		011—方式 3			
				100—方式 4			
				101—方式 5			

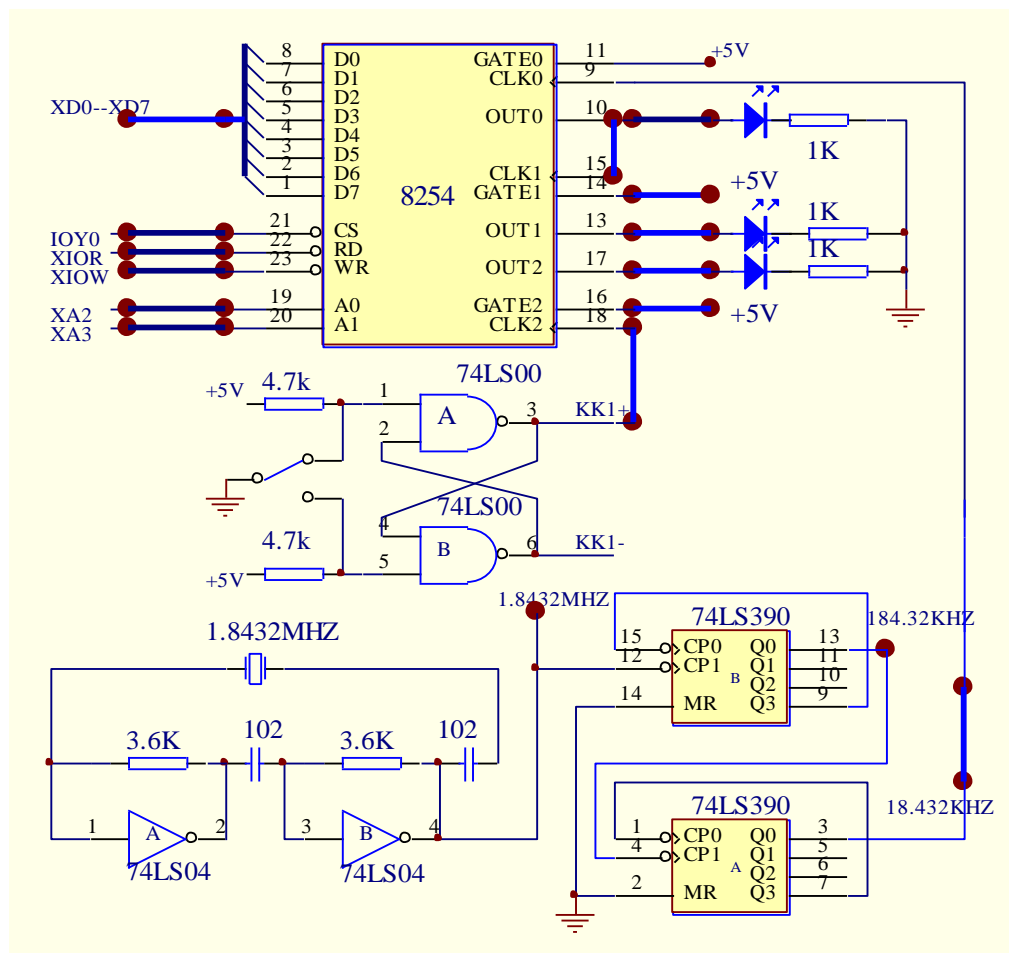
表 3-2-2 8254 读出控制字格式

D7	D6	D5	D4	D3	D2	D1	D0
1	1	0—锁存计数值	0—锁存状态信息	计数器选择（同方式控制字）			0

表 3-2-3 8254 状态字格式

D7	D6	D5	D4	D3	D2	D1	D0
OUT 引脚现行状态 1—高电平 0—低电平	计数初值是否装入 1—无效计数 0—计数有效	计数器方式（同方式控制字）					

8254 定时/计数器电原理图如图 3-2-2 所示，流程图如图 3-2-3 所示。



四. 预习要求

- 1) 熟悉 8254 的编程方法及各种工作方式，补全参考代码中的缺失部分，以达到实验任务中规定的要求。

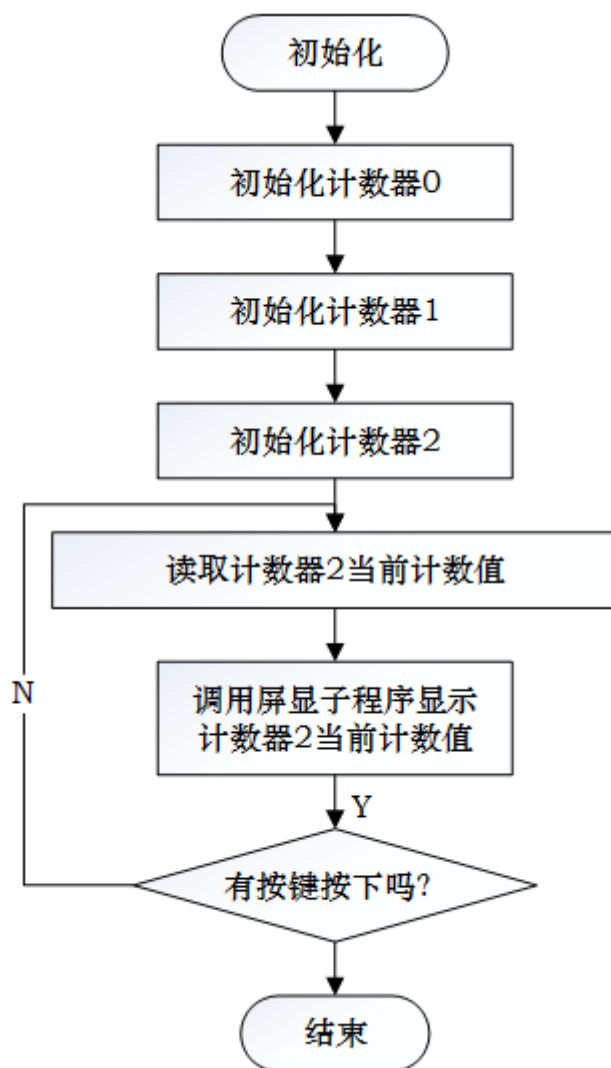



图 3-2-3 8254 定时/计数器参考程序流程图

五. 单元自检

为保证实验单元工作正常，请在实验开始前进行快速单元自检，通过简单的 Debug 命令检查单元硬件是否可以正常工作。参考步骤如下：

1) 参考图 3-2-3 所示连接实验线路。

2) 点击工具栏图标 

3) C:\TangDu\PitPP\Asm>DEBUG

O 300C 36 ; 计数器 0 CW

O 3000 00 ; 计数器 0 低八位初值

O 3000 90 ; 计数器 0 高八位初值,执行后 LED 一灭一亮

计数器 1, 2 自检类似

Q ;退出自检

4) 返回 WINDOWS C:\TangDu\PitPP\Asm>EXIT

六. 参考代码

```
IOY0    EQU    3000H
TIMER0  EQU    IOY0+00H*4    ;8254计数器0端口地址
TIMER1  EQU    IOY0+01H*4    ;8254计数器1端口地址
TIMER2  EQU    IOY0+02H*4    ;8254计数器2端口地址
TCTL    EQU    IOY0+03H*4    ;8254控制寄存器端口地址
```

```
STACK1  SEGMENT STACK
        DW     256 DUP(?)
STACK1  ENDS
```

```
DATA    SEGMENT
MES0    DB     'Pressed:  $'
MES1    DB     'Press any key to exit !' ,0DH,0AH,' $'
NUM      DB     ?
DATA    ENDS
```

```
CODE SEGMENT
        ASSUME  CS:CODE,DS:DATA, SS:STACK1
START:  MOV     AX, DATA
        MOV     DS, AX
        MOV     DX, OFFSET MES1
        MOV     AH, 9
        INT     21H
```

在此补全代码... 完成对计数器0、1、2的初始化

```
L1:     MOV     DX, TIMER2
        IN      AL, DX          } ;读入计数器2值保存
        MOV     NUM,AL
        CALL    DISP
        MOV     AL, NUM
        CMP     AL, 0
        JZ      QUIT           ;计数至0时退出
        MOV     DL, 0FFH
        MOV     AH, 6          } ;判主键盘有无键按下
        INT     21H
        JZ      L1             ; 有键按下跳转

QUIT:   MOV     AX, 4C00H      ;结束程序退出
        INT     21H
```

```

DISP  PROC                                     ;显示子程序
      MOV  DX, OFFSET MES0 }
      MOV  AH, 9           } ; 显示MES0
      INT  21H

      MOV  AL, NUM
      CMP  AL, 9           ;判断是否<=9
      JLE  L2              ;若是则为'0'-'9',ASCII码加30H
      ADD  AL, 7           ;否则为'A'-'F',ASCII码加37H
L2:    ADD  AL, 30H
      MOV  DL, AL
      MOV  AH, 2           } ;在显示器上显示按压开关的次数
      INT  21H
      MOV  DL, 0DH
      INT  21H
      RET
DISP  ENDP

CODE  ENDS
      END    START

```

七. 探究内容

- 1) 尝试改变计数器的工作方式或初值，观察输出的变化；
- 2) 利用实验箱上的其他单元，构造其他的实验方案。例如，输出可控频率的方波至电子发声单元，使其发出预设的声音甚至音乐。

八. 实验报告

- 1) 实验目的和实验内容。
- 2) 通过简要的电原理框图和程序流程图，描述自己设计的实验方案。
- 3) 实验结果。
- 4) 心得体会和建议。

实验三 8259 中断控制器

一. 实验目的

- 1) 学习中断控制器 8259 的工作原理。
- 2) 掌握可编程控制器 8259 的应用编程方法。

二. 实验任务

用单次脉冲单元的 KK1+和 KK2+产生单次脉冲，模拟两个中断源，在 KK1+触发的中断服务程序中显示字符“0”，在 KK2+触发的中断服务程序中显示字符“1”。

三. 原理提要

1. 8259 控制器的介绍

中断控制器 8259A 是 Intel 公司专为控制优先级中断而设计开发的芯片。它将中断源优先级排队、辨别中断源以及提供中断矢量的电路集于一片中，因此无需附加任何电路，只需对 8259A 进行编程，就可以管理 8 级中断，并选择优先模式和中断请求方式，即中断结构可以由用户编程来设定。同时，在不需增加其他电路的情况下，通过多片 8259A 的级连，能构成多达 64 级的矢量中断系统。它的管理功能包括：1) 记录各级中断源请求，2) 判别优先级，确定是否响应和响应哪一级中断，3) 响应中断时，向 CPU 传送中断类型号。8259A 的内部结构和引脚如图 3-3-1 所示。

8259A 的命令共有 7 个，一类是初始化命令字，另一类是操作命令。8259A 的编程就是根据应用需要将初始化命令字 ICW1-ICW4 和操作命令字 OCW1- OCW3 分别写入初始化命令寄存器组和操作命令寄存器组。ICW1-ICW4 的命令字格式如图 3-3-2 所示，OCW1-OCW3 的命令字格式如图 3-3-3 所示，其中 OCW1 用于设置中断屏蔽操作字，OCW2 用于设置优先级循环方式和中断结束方式的操作命令字，OCW3 用于设置和撤销特殊屏蔽方式、设置中断查询方式以及设置对 8259 内部寄存器的读出命令。

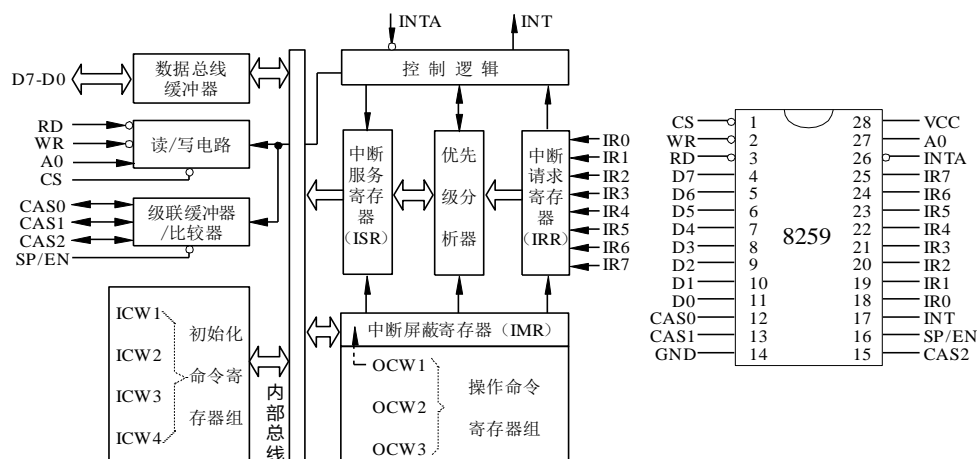


图 3-3-1 8259 内部结构和引脚图

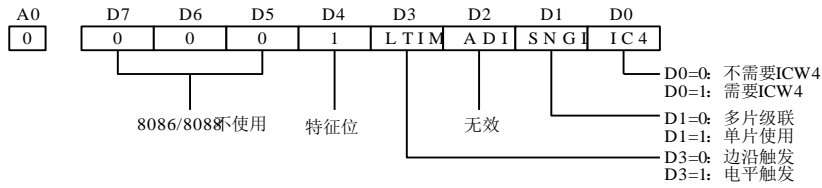


图 3-3-2 (a) ICW1 格式



图 3-3-2 (b) ICW2 格式

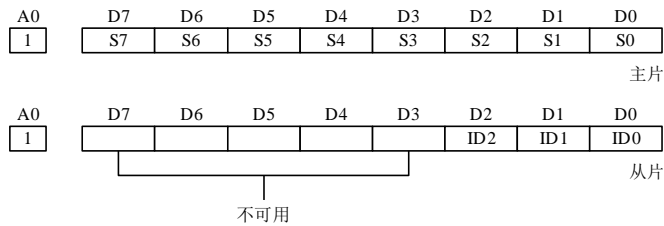


图 3-3-2 (c) ICW3 格式

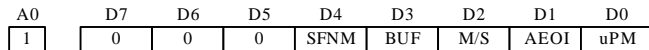


图 3-3-2 (d) ICW4 格式

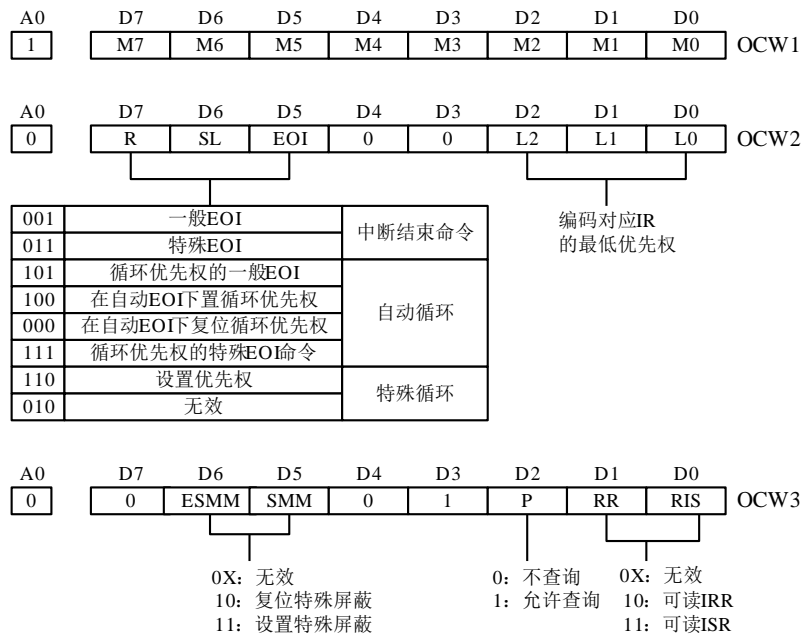


图 3-3-3 OCW 命令字格式

2. 8259 寄存器及命令的控制访问

在硬件系统中，8259 仅占用两个外设接口地址，在片选有效的情况下，利用 A0 来寻址不同的

寄存器和命令字。对寄存器和命令的访问控制如表 3-3-1 所示。

表 3-3-1 8259 寄存器及命令的控制访问

A0	D4	D3	读信号	写信号	片选	操作
0			0	1	0	读出 ISR,IRR 的内容
1			0	1	0	读出 IMR 的内容
0	0	0	1	0	0	写入 OCW2
0	0	1	1	0	0	写入 OCW3
0	1	×	1	0	0	写入 ICW1
1	×	×	1	0	0	写入 OCW1, ICW2, ICW3, ICW4

3. PC 微机系统中的 8259

在 80x86 系列 PC 微机系统中，包含了两片 8259A 中断控制器，经级连可以管理 15 级硬件中断，但其中部分中断号已经被系统硬件占用，具体情况如表 3-3-2 示。两片 8259A 的端口地址为：主片 8259 使用 020H 和 021H 两个端口；从片使用 0A0H 和 0A1H 两个端口。系统初始化两片 8259 的中断请求信号均采用上升沿触发，采用全嵌套方式，优先级的排列次序为 0 级最高，依次为 1 级、8 级~15 级，然后是 3 级~7 级。

在扩展系统总线上的 INTR 对应的中断线就是 PC 机保留中断其中的一个。对 INTR 中断的初始化 PC 机已经完成，在使用时主要是将其中断屏蔽打开，修改中断向量。

表 3-3-2 PC 微机系统中的硬件中断

中断号	功能	中断向量号	中断向量地址
主 8259 IRQ0	日时钟/计数器 0	08H	0020H~0023H
主 8259 IRQ1	键盘	09H	0024H~0027H
主 8259 IRQ2	接从片 8259	0AH	0028H~002BH
主 8259 IRQ3	串行口 2	0BH	002CH~002FH
主 8259 IRQ4	串行口 1	0CH	0030H~0033H
主 8259 IRQ5	并行口 2	0DH	0034H~0037H
主 8259 IRQ6	软盘	0EH	0038H~003BH
主 8259 IRQ7	并行口 1	0FH	003CH~003FH
从 8259 IRQ8	实时钟	70H	01C0H~01C3H
从 8259 IRQ9	保留	71H	01C4H~01C7H
从 8259 IRQ10	保留	72H	01C8H~01CBH
从 8259 IRQ11	保留	73H	01CCH~01CFH
从 8259 IRQ12	保留	74H	01D0H~01D3H
从 8259 IRQ13	协处理器中断	75H	01D4H~01D7H
从 8259 IRQ14	硬盘控制器	76H	01D8H~01DBH
从 8259 IRQ15	保留	77H	01DCH~01DFH

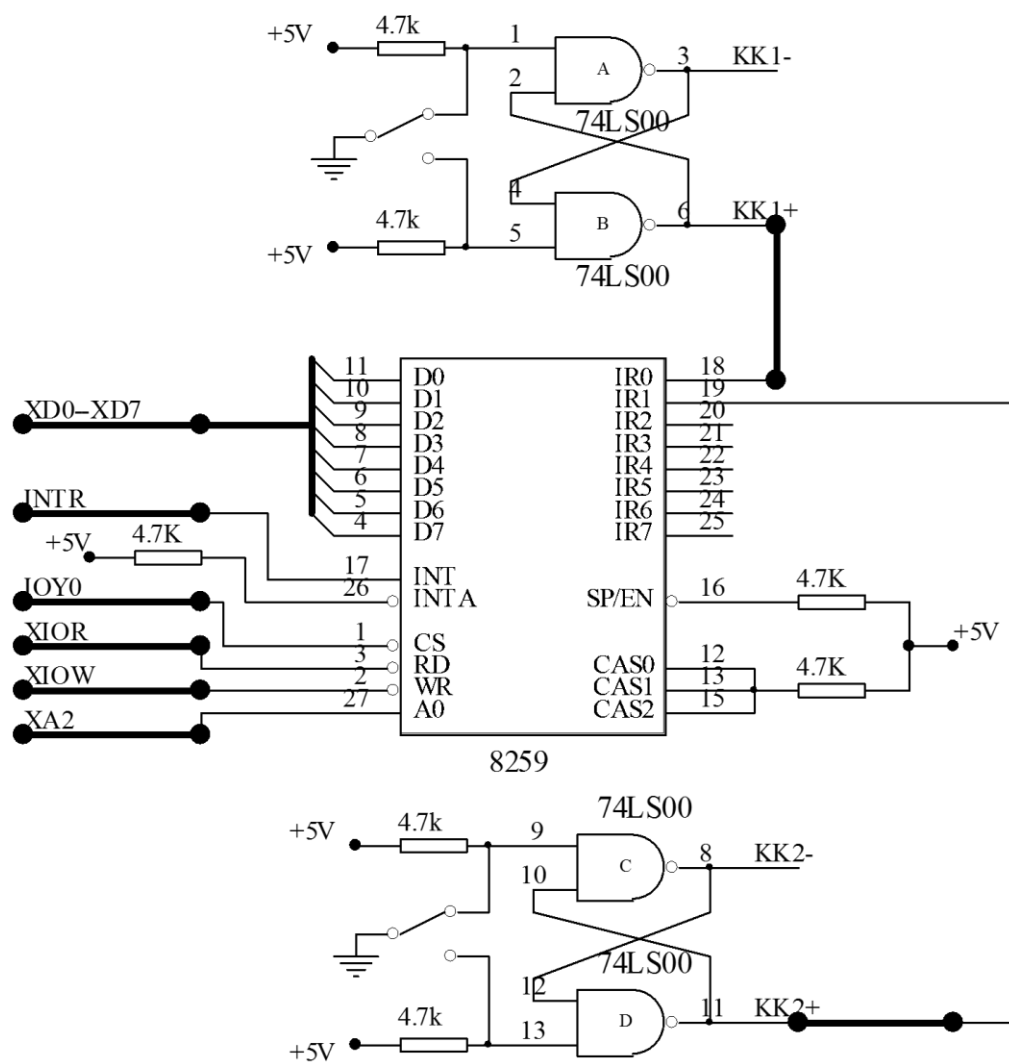


图 3-3-4 8259 中断控制器电原理图

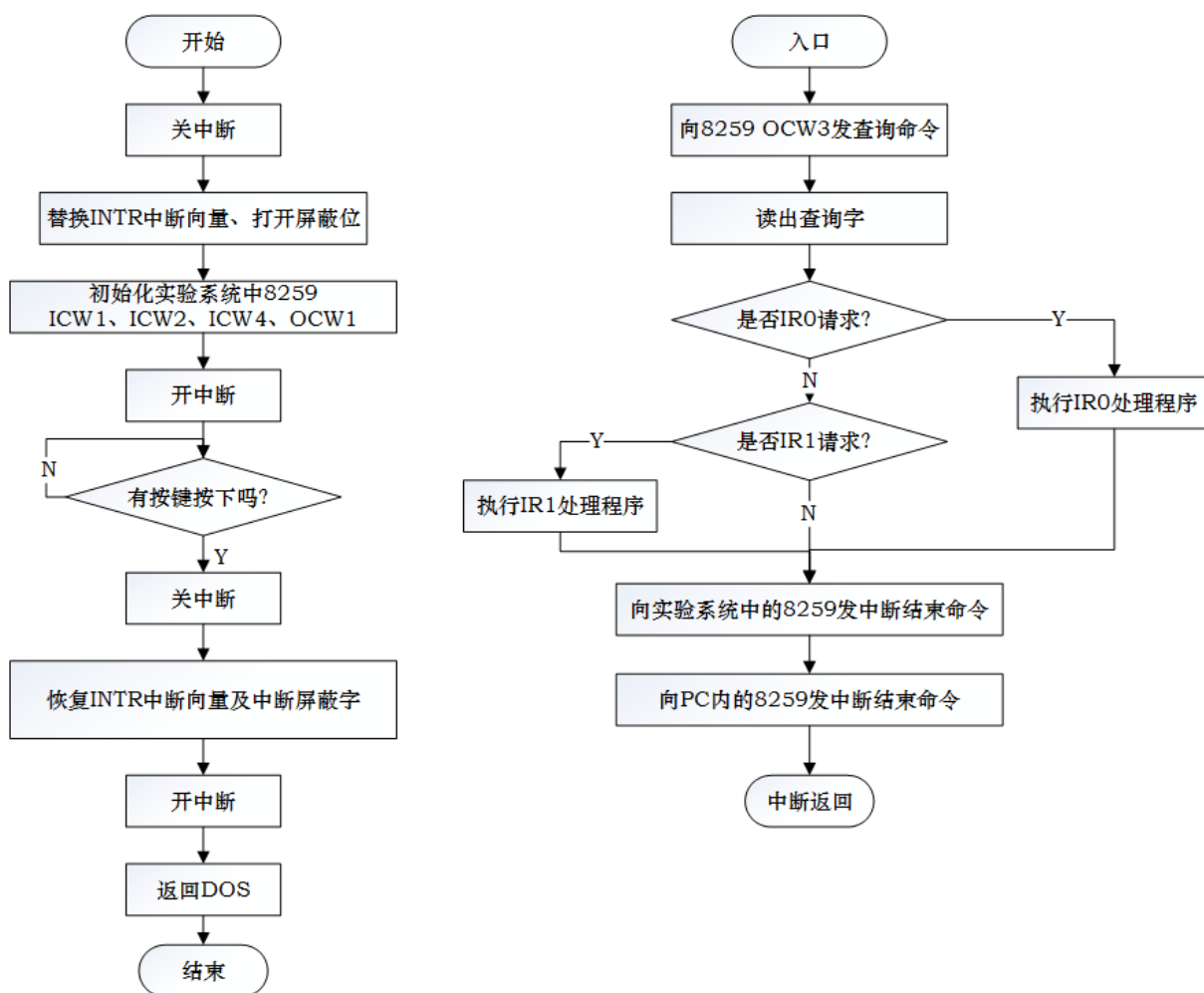


图 3-3-5 8259 中断控制器参考流程图

四. 预习要求

本实验要求实现 8259 控制器 IR0、IR1 两路中断。用 KK1+和 KK2+模拟两个中断源，在 IR0 对应的服务程序中显示字符“0”，在 IR1 对应的服务程序中显示字符“1”。电原理框图如图 3-3-4 所示。实验程序参考流程如图 3-3-5 所示。

将实验箱上的 8259 的 INT 连接到总线单元的 INTR，8259 的 8 路中断请求线 IR0~IR7 就成了单一 INTR 中断请求线的扩充。这 8 路中断源共用 INTR 的中断向量，共用 INTR 的中断服务程序。在 INTR 的中断服务程序中通过对 8259 OCW3 的查询，以确定是 IR0~IR7 中哪个产生中断，然后转到相应的服务线程进行处理。

参照电原理框图和程序流程图，仔细阅读参考代码，充分理解 8259 的编程方法。

五. 参考代码

;8259 扩充中断应用实验

```

INTR_IVADD    EQU    01C8H    ;INTR 对应的中断矢量地址
INTR_OCW1     EQU    0A1H     ;INTR 对应 PC 机内部 8259 的 OCW1 地址
INTR_OCW2     EQU    0A0H     ;INTR 对应 PC 机内部 8259 的 OCW2 地址
  
```

```

INTR_IM      EQU    0FBH    ;INTR 对应的中断屏蔽字
IOY0         EQU    3000H    ;片选 IOY0 对应的端口始地址

MY8259_ICW1  EQU    IOY0+00H ;实验系统中 8259 的 ICW1 端口地址
MY8259_ICW2  EQU    IOY0+04H ;实验系统中 8259 的 ICW2 端口地址
MY8259_ICW3  EQU    IOY0+04H ;实验系统中 8259 的 ICW3 端口地址
MY8259_ICW4  EQU    IOY0+04H ;实验系统中 8259 的 ICW4 端口地址
MY8259_OCW1  EQU    IOY0+04H ;实验系统中 8259 的 OCW1 端口地址
MY8259_OCW2  EQU    IOY0+00H ;实验系统中 8259 的 OCW2 端口地址
MY8259_OCW3  EQU    IOY0+00H ;实验系统中 8259 的 OCW3 端口地址

STACK1 SEGMENT STACK
        DW 256    DUP(?)
STACK1 ENDS
DATA SEGMENT
MES      DB    'Press any key to exit!',0AH,0DH,0AH,0DH,'$'
CS_BAK   DW    ?           ;保存 INTR 原中断处理程序入口段地址的变量
IP_BAK   DW    ?           ;保存 INTR 原中断处理程序入口偏移地址的变量
IM_BAK   DB    ?           ;保存 INTR 原中断屏蔽字的变量
DATA ENDS
CODE SEGMENT
        ASSUME CS:CODE,DS:DATA, SS:STACK1
START:  MOV AX,DATA
        MOV    DS,AX
        MOV    DX,OFFSET MES
        MOV    AH,09H
        INT    21H
        CLI
        MOV    AX,0000H
        MOV    ES,AX
        MOV    DI, INTR_IVADD
        MOV    AX,ES:[DI]
        MOV    IP_BAK,AX
        MOV    AX,OFFSET MYISR
        MOV    ES:[DI],AX
        ADD    DI, 2
        MOV    AX,ES:[DI]
        MOV    CS_BAK,AX
        MOV    AX,SEG MYISR
        MOV    ES:[DI],AX
        MOV    DX,INTR_OCW1
        IN      AL,DX
        MOV    IM_BAK,AL
        AND    AL,INTR_IM
        OUT    DX,AL

```



```

MOV     DX, MY8259_ICW1 } ;初始化实验系统中 8259 的 ICW1
MOV     AL, 13H          } ;边沿触发、单片 8259、需要 ICW4
OUT     DX, AL           }
MOV     DX, MY8259_ICW2 }
MOV     AL, 08H          } ;初始化实验系统中 8259 的 ICW2
OUT     DX, AL           }

MOV     DX, MY8259_ICW4 } ;初始化实验系统中 8259 的 ICW4
MOV     AL, 01H          } ;非自动结束 EOI
OUT     DX, AL           }
MOV     DX, MY8259_OCW3 } ;向 8259 的 OCW3 发送读取 IRR 命令
MOV     AL, 0AH          }
OUT     DX, AL           }
MOV     DX, MY8259_OCW1 } ;初始化实验系统中 8259 的 OCW1
MOV     AL, 0FCH          } ;打开 IR0 和 IR1 的屏蔽位
OUT     DX, AL           }
STI

WAIT1:  MOV     AH, 1                ;判断是否有按键按下
        INT     16H
        JZ      WAIT1              ;无按键则跳回继续等待，有则退出

QUIT:   CLI
        MOV     AX, 0000H           ;恢复 INTR 原中断矢量
        MOV     ES, AX
        MOV     DI, INTR_IVADD     } ;恢复 INTR 原中断处理程序入口偏移地址
        MOV     AX, IP_BAK
        MOV     ES:[DI], AX
        ADD     DI, 2
        MOV     AX, CS_BAK         } ;恢复 INTR 原中断处理程序入口段地址
        MOV     ES:[DI], AX
        MOV     DX, INTR_OCW1     } ;恢复 INTR 原中断屏蔽寄存器的屏蔽字
        MOV     AL, IM_BAK
        OUT     DX, AL
        STI
        MOV     AX, 4C00H          ;返回到 DOS
        INT     21H

MYISR PROC NEAR                ;中断处理程序 MYISR
        PUSH    AX
QUERY:  MOV     DX, MY8259_OCW3    ;向 8259 的 OCW3 发送读取 IRR 命令
        IN      AL, DX             ;读出 IRR 寄存器值

        AND     AL, 03H
        CMP     AL, 01H
        JE      IR0ISR             ;若为 IR0 请求，跳到 IR0 处理程序

```

```

        JNE      IR1ISR      ;若为 IR1 请求，跳到 IR1 处理程序
        JMP      OVER

IROISR:MOV     AL, 30H
        MOV      AH, 0EH
        INT      10H
        MOV      AL, 20H
        INT      10H
        JMP      OVER      } ;IR0 处理，显示字符串 STR0

IR1ISR:MOV     AL, 31H
        MOV      AH, 0EH
        INT      10H
        MOV      AL, 20H
        INT      10H
        JMP      OVER      } ;IR1 处理，显示字符串 STR1

OVER:  MOV     DX,INTR_OCW2
        MOV     AL, 20H
        OUT     DX,AL
        MOV     AL, 20H
        OUT     20H, AL
        POP     AX
        IRET

MYISR ENDP

CODE ENDS
        END START

```

六. 实验报告

- 1) 实验目的和实验内容。
- 2) 通过简要的电原理框图和程序流程图，描述自己设计的实验方案。
- 3) 实验结果。
- 4) 心得体会和建议。

实验四 8255 并口控制器

一. 实验目的

- 1) 掌握 8255 的工作方式及应用编程。
- 2) 学习键盘扫描的原理及电路接法。
- 3) 掌握利用 8255 实现按键扫描及数码管显示方法。

二. 实验任务

用 8255 实现键盘扫描与显示功能,当小键盘有键按下时,键值在数码管最右位的位置上显示,数码管上显示最新六位键值内容,当主键盘有键按下时,返回系统。

三. 原理提要

8255 可编程接口芯片是 Intel 公司生产的通用并行 I/O 接口芯片,它具有 A、B、C 三个并行接口,用+5V 单电源供电,能在以下三种方式下工作:

方式 0--基本输入/出方式。

方式 1--选通输入/出方式。

方式 2--双向选通工作方式。

8255 的内部结构及引脚如图 3-4-1 所示,8255 工作方式控制字和 C 口置位/复位控制字格式如图 3-4-2 所示。

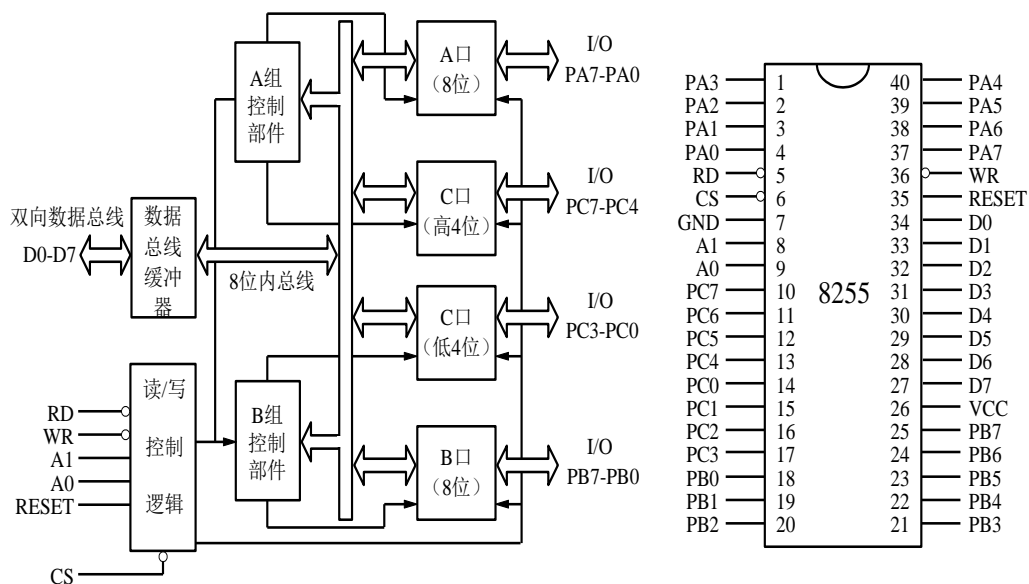


图 3-4-1 8255 的内部结构及引脚

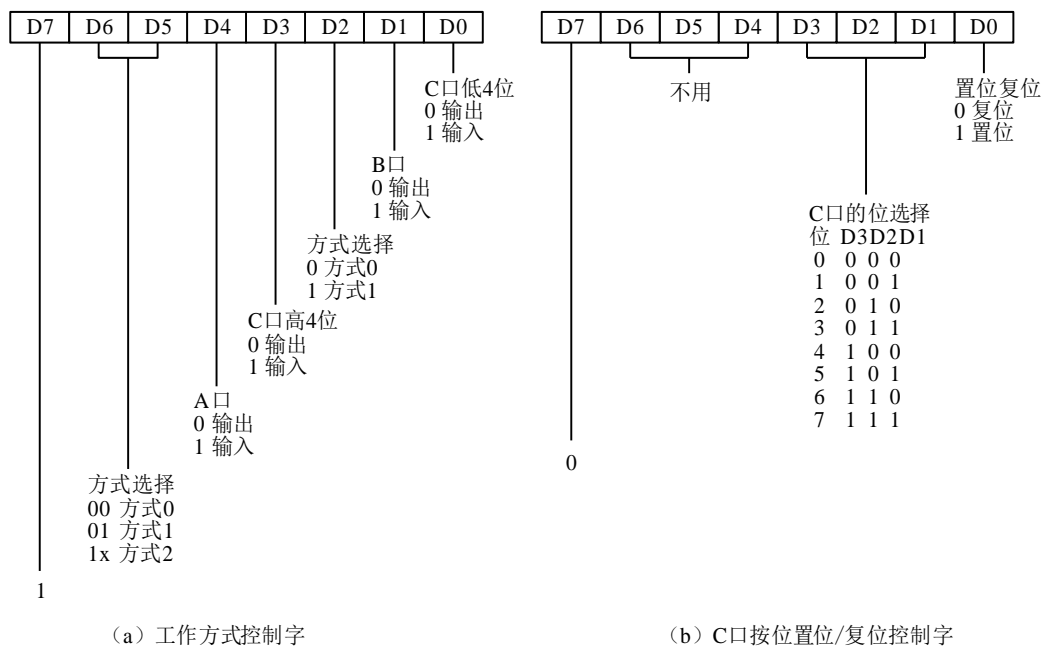


图 3-4-2 8255 控制字格式

在本实验中，键盘是由 4 行×4 列共 16 个按键组成，其中 PC0~PC3 作为行选择信号 Y1~Y4,PC4~PC7 作为列扫描信号 X1~X4。在键盘扫描中,常用的方法有逐行扫描和行反转法两种,本实验采用行反转法,其键盘扫描方式如下：

1. 写 8255 工作方式控制字,使 C4~PC7 为 0(列值为 0),然后读入 PC0~PC3(行值),如果此时有键按下,则对应的行值为 0。
2. 写 8255 工作方式控制字, 使 PC0~PC3 为 0(行值为 0),再读入 C4~PC7(列值),因此时按键没有松开,故对应的列值为 0;因此,当一个键按下时,可以读到一对唯一的行值和列值
3. 合并行值和列值,与每个键盘扫描码逐一比较,就可以确定是那一个键按下了。

在软件设计上要注意消除按键抖动的处理以及数码管显示的刷新。

数码管显示器是由 6 位共阴的 7 段数码管组成,在电路中它也接成扫描电路方式，6 位共阴的 7 段数码共用段码 A~G、Dp,当位码 L1~L6 某位 0 为时,则该位数码管点亮,显示相应的字符。要在六位数码管上显示不同的内容,需一定频率循环地往六位数码管上输送段码与位码,在因此这种显示方式称为动态显示方式。

键盘扫描及显示电原理图如图 3-4-3 所示，主程序流程如图 3-4-4 所示，键盘扫描子程序流程图如图 3-4-5 所示，显示子程序流程图如图 3-4-6 所示

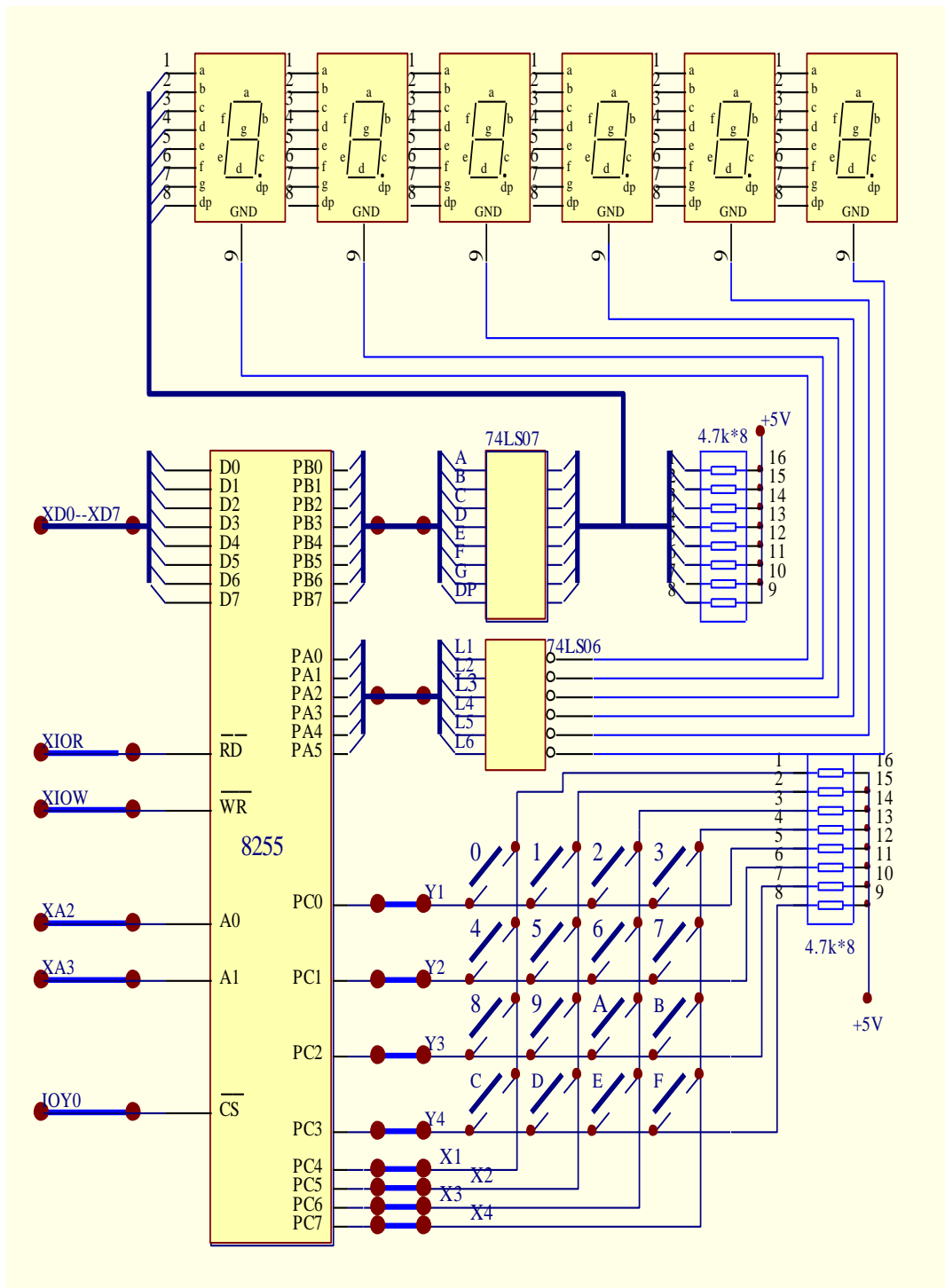


图 3-4-3 键盘扫描及显示实验电原理框图

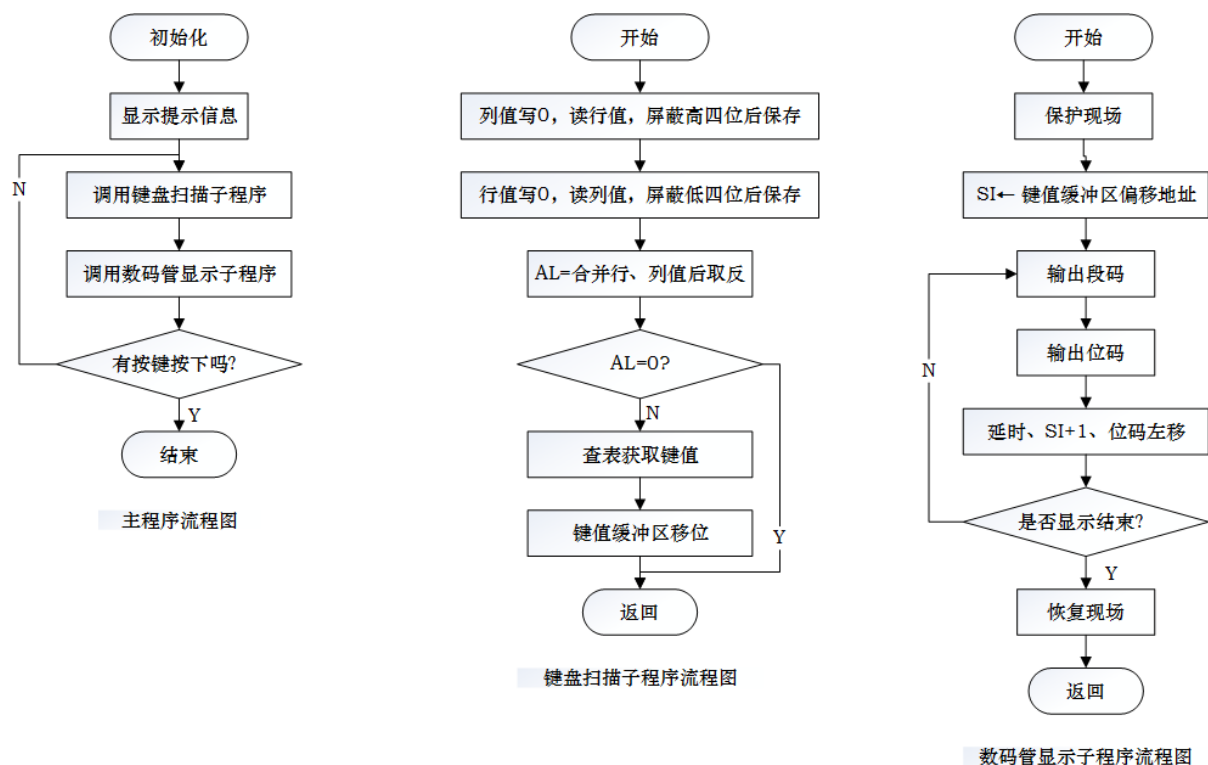


图 3-4-4 主程序、键盘扫描及显示子程序流程图

四. 预习要求

- 1) 熟悉 8255 的基本编程方法;
- 2) 参照电原理框图和程序流程图, 仔细阅读参考代码, 理解键盘扫描的基本原理, 以及驱动数码管显示键值的基本方法。
- 3) 参考代码中取反后的键盘扫描码是如何得出的?

五. 单元自检

为保证实验单元工作正常, 请在实验开始前进行快速单元自检, 通过简单的 Debug 命令检查单元硬件是否可以正常工作。参考步骤如下:

- 1) 参考图 3-4-4 所示连接实验线路。

- 2) 点击工具栏图标 

- 3) C:\TangDu\PitPP\Asm>DEBUG

```

O 300C 81      ;8255 写 CW,A 口、B 口方式 0 输出;C 下口输入, C 上口输出
O 3004 3F      ; B 口输出 “0” 的段码
O 3000 1       ; A 口输出 1,数码管左第 1 位显示 0
O 3008 0F      ; C 口列输出为 0
I 3008         ;当按压 “0” 键时,读入的值为 0E
O 300C 88      ;8255 写 CW,A 口、B 口方式 0 输出;C 上口输入, C 下口输出
O 3008 F0      ; C 口输出行为 0
  
```

```

I    3008          ;当按压“0”键时,读入的值为 E0
Q          ;退出自检

```

4) 返回 WINDOWS C:\TangDu\PitPP\Asm>EXIT

六. 参考代码

;键盘扫描及数码管显示实验

```

IOY0    EQU    3000H          ;片选IOY0对应的端口始地址
PA55    EQU    IOY0+00H*4      ;8255的A口地址
PB55    EQU    IOY0+01H*4      ;8255的B口地址
PC55    EQU    IOY0+02H*4      ;8255的C口地址
PCTL    EQU    IOY0+03H*4      ;8255的控制寄存器地址
DATA    SEGMENT
BUFF    DB      6 DUP (10H)
TABLE1 DB      11H,21H,41H,81H,12H,22H,42H,82H      ;取反后的键盘扫描码
        DB      14H,24H,44H,84H,18H,28H,48H,88H
DCTBL   DB      3Fh,06h,5Bh,4Fh,66h,6Dh,7Dh,07h,7Fh,6Fh } ;数码管的段码表
        DB      77h,7Ch,39h,5Eh,79h,71h,00H
MES      DB      'Press any key on the small keyboard!',0DH,0AH
        DB      'Press key to display on the led!',0dh,0ah, '$'
MESS     DB      'Press main keyboard any key to exit! ',0DH,0AH,0DH,0AH,'$'
KEYC     DB      ?
KEY      DB      ?
DATA     ENDS
STAC     SEGMENT PARA STACK
        DB      256 DUP(?)
STAC     ENDS
CODE     SEGMENT
        ASSUME CS:CODE,DS:DATA,SS:STAC
START:   MOV     AX, DATA
        MOV     DS, AX
        MOV     DX, OFFSET MES      } ;显示MES
        MOV     AH, 9
        INT     21H
        MOV     DX, OFFSET MESS     } ;显示MESS
        MOV     AH, 9
        INT     21H
LOP1:    CALL    TESTKEY
        CALL    DISP
        MOV     DL, 0FFH
        MOV     AH, 6
        INT     21H
        JZ      LOP1
QUIT:    MOV     AX, 4C00H
        INT     21H
TESTKEY PROC

```

```

KEY0:  MOV    AL, 81H
        MOV    DX, PCTL
        OUT    DX, AL
                                } ;8255控制字,PC0-3入,PC4-7出

        MOV    AL, 00
        MOV    DX, PC55
        OUT    DX, AL
                                } ; C 口输出0
        IN     AL, DX
        AND    AL, 0FH
                                } ;读入行值,屏蔽列值后保存
        MOV    KEYC,AL

KEY1:  MOV    AL, 88H
        MOV    DX, PCTL
        OUT    DX, AL
                                } ;8255控制字,PC0-3出,PC4-7入
        MOV    AL, 00
        MOV    DX, PC55
        OUT    DX, AL
                                } ; C 口输出0
        IN     AL, DX
        AND    AL, 0F0H
                                } ; 读入列值,屏蔽行值后合并取反
        OR     AL, KEYC
        NOT    AL
        CMP    AL,0
                                ;无键按下退出子程序
        JZ     KEYEND
        MOV    SI, OFFSET TABLE1
        MOV    CX, 16
        MOV    DL, 00H
        KEY2: CMP    AL, [SI]
        JZ     KEY3
        INC    SI
        INC    DL
        DEC    CX
        JZ     KEYEND
        JMP    KEY2
                                } ;查找按键的值

KEY3:  MOV    KEY, DL
        MOV    SI, OFFSET BUFF+1
        MOV    DI, OFFSET BUFF
        MOV    CX, 5
        KEY4: MOV    AL, [SI]
        MOV    [DI],AL
        INC    SI
        INC    DI
        LOOP   KEY4
                                } ;显示缓冲区内内容向前移一位
        MOV    AL, KEY
                                ;当前键值存入BUF[5]单元
        MOV    [DI],AL

```



```

        MOV     AL, 88H
        MOV     DX, PCTL
        OUT     DX, AL
KEY5:    MOV     AL, 00
        MOV     DX, PC55
        OUT     DX, AL
        IN      AL, DX
        AND     AL, 0F0H
        CMP     AL, 0F0H
        JNZ     KEY5
KEYEND:  RET
TESTKEY ENDP

```

} ; 8255控制字,PC0-3出,PC4-7入
 } ;判断按键是否释放

```

DISP    PROC
        PUSH    DS
        PUSH    AX

        MOV     CL, 1
        MOV     SI, OFFSET  BUFF
DIS2:   MOV     AL, [SI]
        LEABX, DCTBL
        XLAT
        MOV     DX, PB55
        OUT     DX, AL

        MOV     DX, PA55
        MOV     AL, CL
        OUT     DX, AL
        CALL    DELAY
        INC     SI
        ROL     CL, 1
        CMP     CL, 40H
        JNZ     DIS2
        POP     AX
        POP     DS
        RET
DISP    ENDP

```

} ;输出段码
 } ;输出位码
 ;段码地址+1
 ;位码向左移1位
 ; 位码是最后位吗?
 ;不是最后位,转DIS2
 ;是最后位,返回

```

DELAY PROC NEAR
        PUSH    CX
        MOV     BX, 30H
DEL1:   MOV     CX, 0FFFFH
        LOOP    $
        DEC     BX
        JNZ     DEL1

```

} ;延时子程序

```
        POP    CX
        RET
DELAY ENDP

CODE ENDS
        END    START
```

七. 探究内容

- 1) 尝试改变参考程序中预设的键值，按下小键盘后在数码管上显示其他内容。
- 2) 尝试改变实验方案中的输出方式。例如，利用实验箱上的“点阵LED显示单元”，用8255的A口和B口控制其显示更为丰富的内容。

八. 实验报告

- 1) 实验目的和实验内容。
- 2) 通过简要的电原理框图和程序流程图，描述自己设计的实验方案。
- 3) 实验结果。
- 4) 心得体会和建议。

实验五 A/D 与 D/A 转换实验

一. 实验目的

- 1) 学习掌握模/数,/数模信号转换基本原理。
- 2) 掌握 ADC0809,0832 芯片的使用方法。

二. 实验任务

从键盘输入两位十六进制值,经 0832 转换成数字量后,送 0809 的通道 0,再经 A/D 转换后在屏幕上显示此数字量。

三. 原理提要

ADC0809 包括一个 8 位的逐次逼近型的 ADC 部分,并提供一个 8 通道的模拟多路开关和联合寻址逻辑。用它可直接输入 8 个单端的模拟信号,分时进行 A/D 转换,在多点巡回检测、过程控制等应用领域中使用非常广泛。ADC0809 的主要技术指标为:

- 分辨率: 8 位 · 单电源: +5V
- 总的不可调误差: $\pm 1\text{LSB}$ · 转换时间: 取决于时钟频率
- 模拟输入范围: 单极性 0~5V · 时钟频率范围: 10KHz~1280KHz

ADC0809 的外部管脚如图 3-5-1 所示,地址信号与选中通道的关系如表 3-5-1 所示。

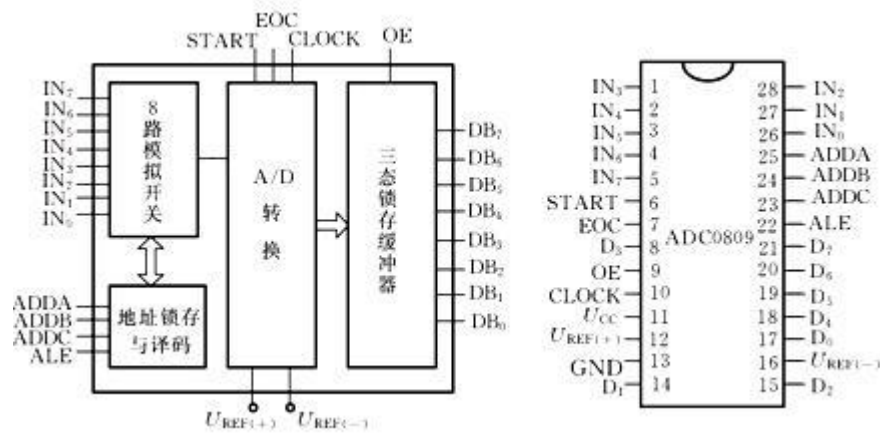


图 3-5-1 ADC0809 管脚图

表 3-5-1 地址信号与选中通道的关系

C B A	选中通道
000	IN0
001	IN1
010	IN2
011	IN3

100	IN4
101	IN5
110	IN6
111	IN7

该实验中是用延时方式读取转换结果的，它是在 0809 启动后，用定时时间大于 A/D 转换时间后，直接读取转换结果，而不利用 EOC 信号。

D/A 转换器是一种将数字量转换成模拟量的器件，其特点是：接收、保持和转换的数字信息，不存在随温度、时间漂移的问题，其电路抗干扰性较好。大多数的 D/A 转换器接口设计主要围绕 D/A 集成芯片的使用及配置响应的外围电路。DAC0832 是 8 位芯片，采用 CMOS 工艺和 R-2RT 形电阻解码网络，转换结果为一对差动电流 Iout1 和 Iout2 输出。

DAC0832 的外部管脚如图 3-5-2 所示，性能参数如表 3-5-2 所示。

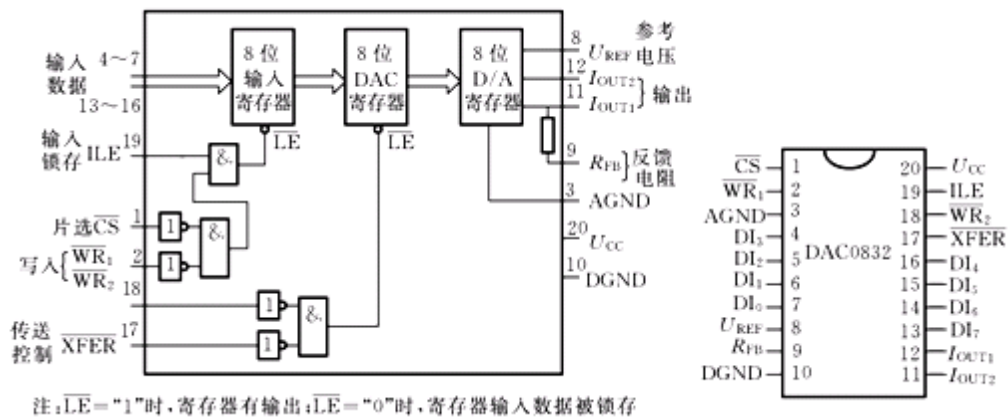


图 3-5-2 DAC0832 的引脚图

表 3-5-2 DAC0832 性能参数

性能参数	参数值
分辨率	8 位
单电源	+5V~ +15V
参考电压	+10V~-10V
转换时间	1Us
满刻度误差	± 1LSB
数据输入电平	与 TTL 电平兼容

A/D 与 D/A 转换实验电原理框图如图 3-5-3 所示，程序流程图如图 3-5-4 所示。

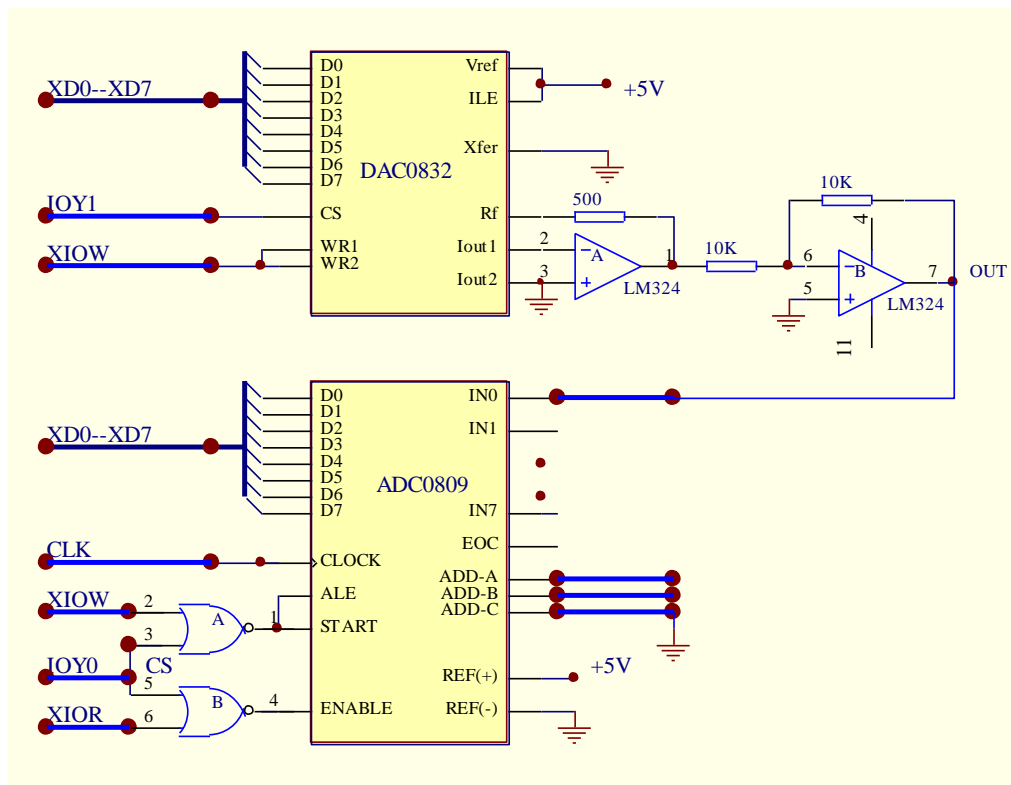


图 3-5-3 A/D 转换实验电原理框图

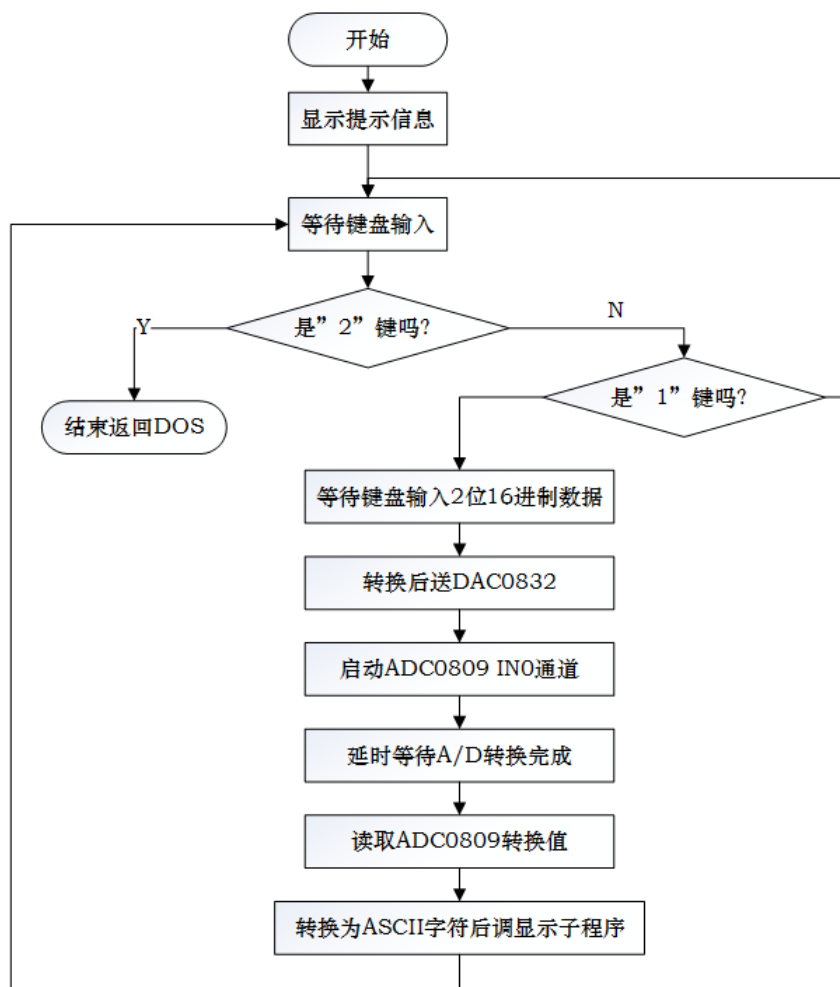


图 3-5-4 A/D 与 D/A 转换实验参考程序流程图

四. 预习要求

- 1) 熟悉 DAC0832 和 ADC0809 两块芯片的基本功能；参照电原理框图和程序流程图，仔细阅读参考代码，理解该实验方案的基本原理。
- 2) 注意图 3-5-3 中 ADC0809 的接线方式，结合参考代码，理解如何启动 ADC0809 的某通道进行模数转换？
- 3) 注意分辨率、转换时间、精度等参数，在实际应用中对芯片选型的影响。

五. 单元自检

- 1) 参考图 3-5-2 所示连接实验线路。

- 2) 点击工具栏图标 

- 3) C:\TangDu\PitPP\Asm>DEBUG

O 3040 80 ;0832 输出 80H,用示波器或万用表测量应为 2.5V

O 3000 00 ;启动 0809 的 IN0

I 3000 ;读入 0809 IN0 值

Q

- 4) 返回 WINDOWS C:\TangDu\PitPP\Asm>EXIT

六. 参考代码

```
CRLF  MACRO
        MOV     DL,  0DH
        MOV     AH,  02H
        INT     21H
        MOV     DL,  0AH
        INT     21H
        ENDM
        } ;宏定义了回车,换行

IOY0  EQU  3000H    ;片选IOY0对应的端口始地址
IOY1  EQU  3040H    ;片选IOY1对应的端口始地址
ADCS  EQU  IOY0     ;AD0809的端口地址
DACS  EQU  IOY1     ;DAC0832的端口地址
STACK1 SEGMENT STACK
        DW  256  DUP(?)
STACK1  ENDS
DATA  SEGMENT
MES0  DB      'PRESS 1 TO INPUT DATA!',0DH,0AH
        DB      'PRESS 2 TO QUIT!',0DH,0AH,0DH,0AH,'$'
MES1  DB      '*****PLEASE INPUT DATA OF HEX!*****',0DH,0AH,'$'
```

```

MES2 DB      '0832 OUTPUT DATA =  $'
MES3 DB      '0809 INPUT DATA =  $'
BUF  DB      2 DUP (?)
DATA  ENDS
STAC  SEGMENT PARA STACK
      DB      256 DUP(?)
STAC  ENDS
CODE  SEGMENT
      ASSUME  CS:CODE,SS:STAC,DS:DATA
START: MOV     AX, DATA
      MOV     DS, AX
      LEA     DX, MES0
      MOV     AH, 9
      INT     21H
      } ;显示MES0
LOP1:  MOV     DL, 0FFH
      MOV     AH, 6
      INT     21H
      } ;检测键盘输入
      JZ      LOP1
      CMP     AL, '1'
      JZ      DA
      CMP     AL, '2'
      JZ      EXIT0
      JMP     START
EXIT0: JMP     EXIT
DA:    LEA     DX, MES1
      MOV     AH, 9
      INT     21H
      } ;显示MES1
      LEA     DX, MES2
      MOV     AH, 9
      INT     21H
      } ;显示MES2
      MOV     AH, 1
      INT     21H
      MOV     BUF, AL
      INT     21H
      MOV     BUF[1], AL
      MOV     AH, 2
      CRLF
      } ;十六进制值存入BUF和BUF[1]
DA0:   MOV     AL, BUF
      SUB     AL, 30H
      CMP     AL, 9
      JBE     A0
      SUB     AL, 7
      } ;十六进制转换十进制
A0:    MOV     BL, AL
      MOV     AL, BUF[1]
      SUB     AL, 30H
      } ;十六进制转换十进制

```

```

        CMP    AL, 9
        JBE    B0
        SUB     AL, 7
B0:     MOV    CL, 4
        ROL    BL, CL
        XOR    AL, BL
        MOV    DX, DACS
        OUT    DX, AL

AD:     MOV    DX, ADCS           ;启动0809 IN0
        OUT    DX, AL
        CALL   DELAY
        LEA    DX, MES3
        MOV    AH, 9             } ;显示MES2
        INT    21H
        MOV    DX, ADCS
        IN     AL, DX            } ;读入0809 IN0值
        MOV    BL, AL
        AND    AL, 0F0H
        MOV    CL, 4             } ; 显示高位
        ROR    AL, CL
        CALL   CRT1
        MOV    AL, BL
        AND    AL, 0FH          } ;显示低位
        CALL   CRT1
        CRLF
        INT    21H
        JMP    START
EXIT:    MOV    AX, 4C00H
        INT    21H

CRT1     PROC
        ADD    AL, 30H
        CMP    AL, 39H
        JBE    D0
        ADD    AL, 7
D0:     MOV    DL, AL
        MOV    AH, 2
        INT    21H
        RET
CRT1     ENDP

DELAY     PROC    NEAR
        PUSH    CX
        MOV     CX, 0FFFFH

```



```

        LOOP    $                ;延时
        POP     CX
        RET
DELAY   ENDP
CODE    ENDS
        END     START

```

七. 探究内容

- 1) 假设需要周期性的定时对某模拟信号持续采样，进行AD转换后输出，利用实验箱上的现有资源设计实验方案。

八. 实验报告

- 1) 实验目的和实验内容。
- 2) 通过简要的电原理框图和程序流程图，描述自己设计的实验方案。
- 3) 实验结果。
- 4) 心得体会和建议。

第四章 32 位微机接口课程设计

课程设计一 数据采集系统一

一. 课设目的:

进一步掌握微机原理知识, 了解微机在实时采集过程中的应用, 学习、掌握编程和程序调试方法。

二. 仪器设备:

微机, 微机接口实验箱, 示波器, 三用表等。

三. 课设内容和要求:

用查询法, 将 ADC 0809 通道 0 外接 0 ~ 5V 电压, 转换成数字量后, 送 DAC0832 输出; 同时在七段 LED 数码管上, 以小数点后两位 (毫伏) 的精度, 显示其模拟电压的十进值; 调整电位器, 用示波器或三用表观察 DAC 0832 的变化, 观察七段 LED 数码管上数值的变化。

要有较好的人机对话界面; 用并行口 8255 的外接小键盘控制程序的运行, 若 “A” 键按下时, 开始数据采集, 在数据采集过程中, 若主键盘有键按下, 则停止运行, 等待 8255 小键盘输入, 当键值是 “E” 时, 返回 DOS; 当按键是 “A” 时, 再次数据采集, 其他键则等待。ADC 0809 的 CLK 脉冲, 由定时器 8254 的 OUT0 提供; ADC 0809 的 EOC 信号, 用 8255 的 PA7 检测。

四. 总体设计

- 1) ADC 0809 的 IN0 采集电位器 0 — 5V 电压。
- 2) DAC 0832 将 ADC 0809 的 IN0 数字量重新转换成模拟量后输出, 供示波器或三用表检测。
- 3) 8255 用于检测 ADC 0809 转换是否结束。
- 4) 8255 用于控制程序的开始和结束。
- 5) 七段 LED 数码管显示 ADC 0809 的 IN0 的值。
- 6) 8254 提供 ADC 0809 的采样时钟脉冲, $F=600\text{KHZ}$ 。

五. 硬件设计

因采用了 PC 机和微机实验箱, 硬件电路设计相对比较简单, 主要利用微机实验箱上的 8255 并行口、ADC 0809、DAC 0832、七段 LED 数码管单元、8254 定时/计数器、电位器等单元电路, 就构成了数据采集系统, 硬件电原理框图 4-1-1 所示。

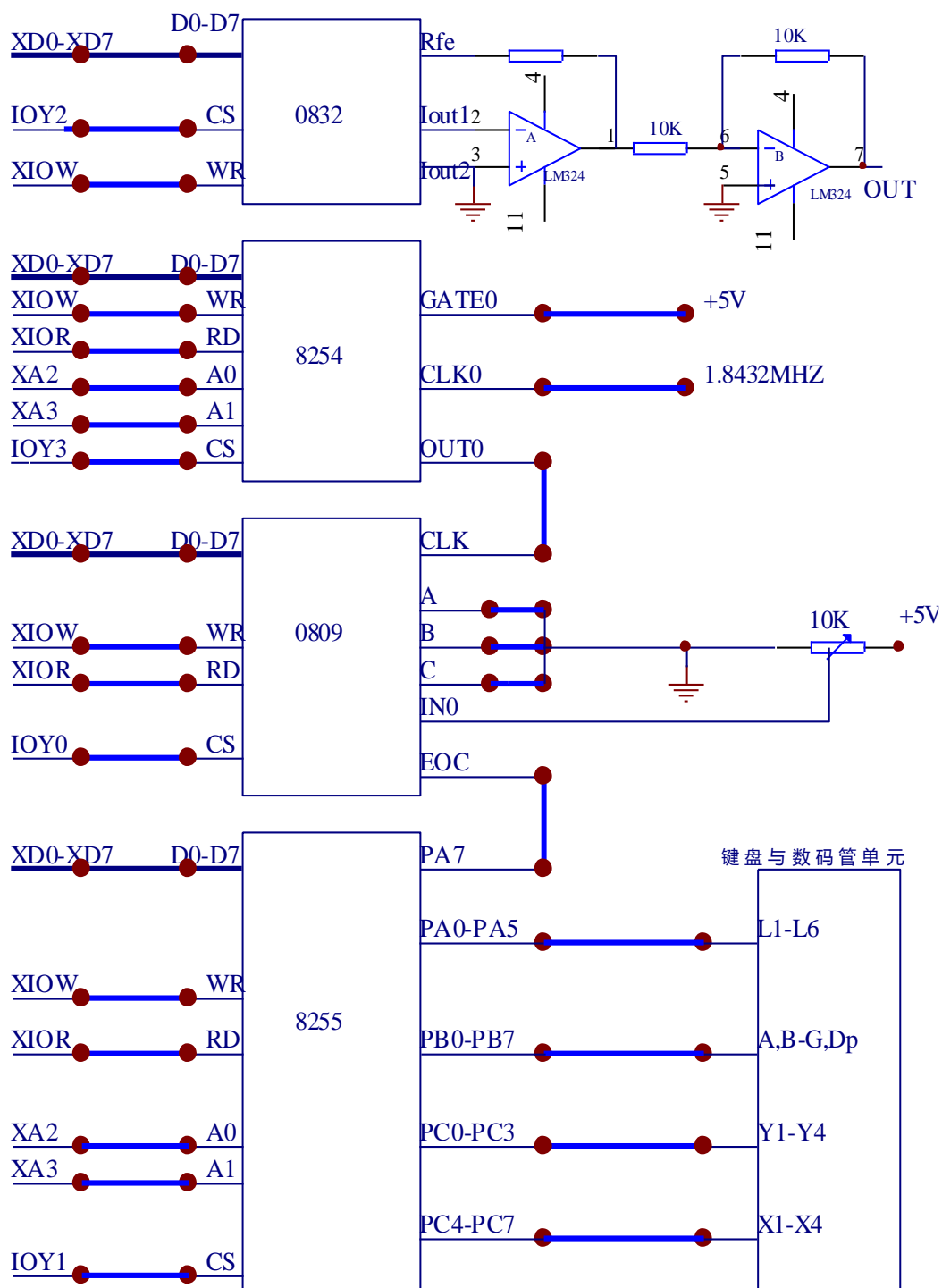


图 4-1-1 数据采集系统参考电原理框图

六. 软件设计

本设计通过软件编程，实现数据采集，0809 由对 IN0 0-5V 直流电压的采样，并将 IN0 的值转换成十进制后，在七段数码管上显示。

1、设计思想

数据采集系统分成四个功能模块，分别是键盘扫描模块、A/D 转换和 D/A 转模块、

量纲转换模块、数码管显示模块。

(1) 键盘扫描模块

键盘扫描模块可参考 8255 并行口实验。

(2) A/D 转换和 D/A 转换模块

根据题目要求,采用查询方式实现模/数转换,在 0809 IN0 开始启动后,程序不断地检查 8255 的 PC7 位,直到高电平,就读取 IN0 的值并保存。此数字量分成两部分输出,一部分送给 0832 输出,以供示波器或三用表检查;另一部分将此数字量转换成十进制数后,分别送到个位、十分位、百分位存储单元保存,以供七段数码管显示时调用。

(3) 量纲转换模块

量纲转换模块简单,只要将 IN0 的数字量分别除以 51,商存入个位存储单元;余数乘以十,再除以 51,商存入十分位存储单元;余数再乘以十,除以 51,若余数大于 25,则商加 1,小于 25 则舍去,这样就达到四舍五入的精度,再把商存入百分位存储单元;以供显示时调用。

(4) 数码管显示模块

数码管显示模块比较简单,应注意的是在显示个位时,要加上小数点的显示,这可以在查表获取个位段码后,再加上 80H 来实现,注意每一位显示后要适当延迟时间。

2. 流程图

数据采集主程序流程图如图 4-1-2 所示;键盘扫描子程序如图 2-4-5C 所示;数码管显示子程序流程图如图 2-4-5B 所示。

七. 课程设计报告

- 1) 课程设计目的和内容。
- 2) 硬件设计:电原理框图及电原理框图工作过程的简要说明。
- 3) 软件设计:程序流程图和接口部件的程序段。
- 4) 编程和调试中遇到什么问题,是怎样解决的。
- 5) 运行结果,体会和建议。
- 6) 程序清单。

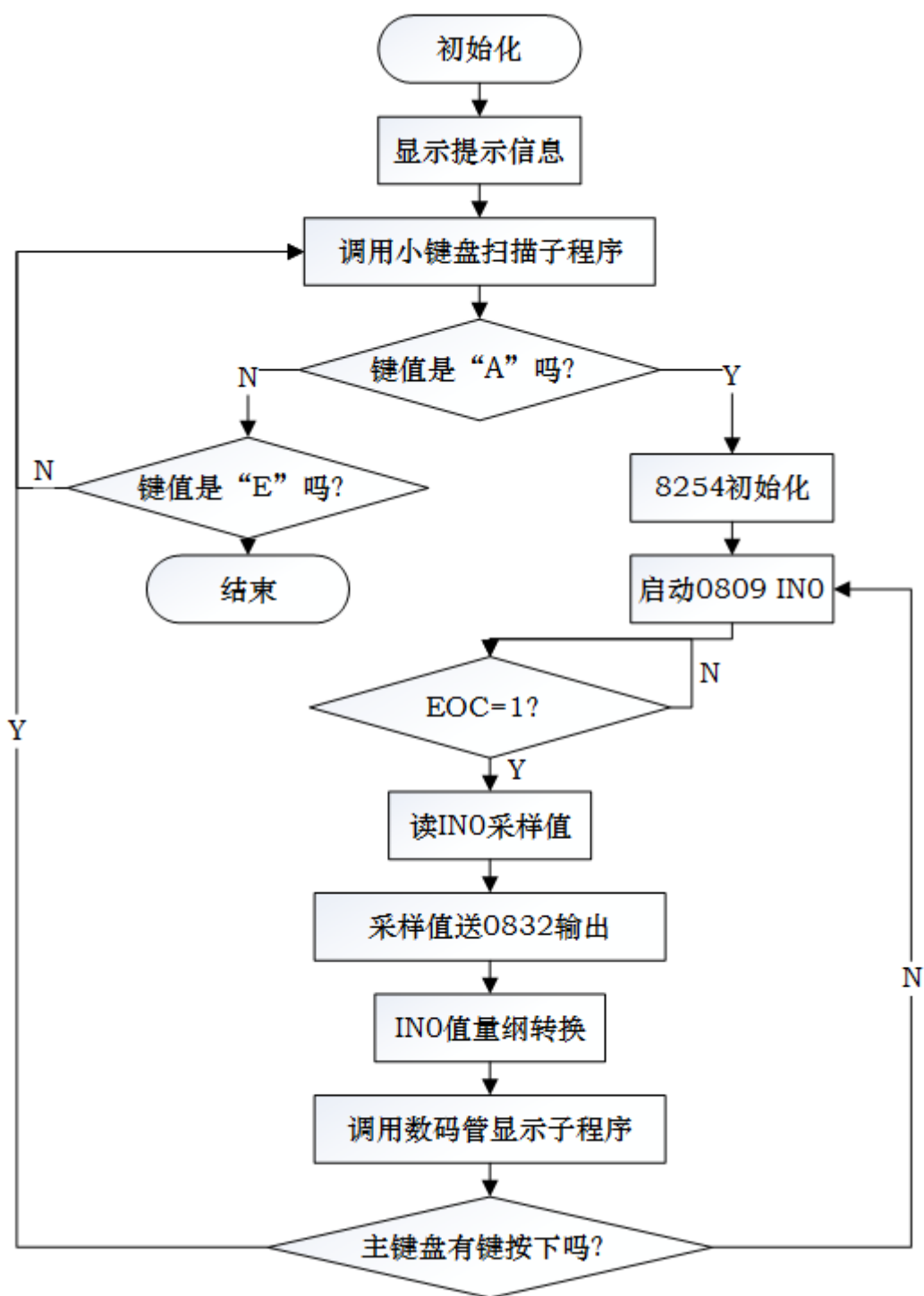


图 4-1-2 数据采集主程序流程图

课程设计二 数据采集系统二（查询法）

一. 课设目的

进一步掌握微机原理知识，了解微机在实时采集过程中的应用，学习、掌握编程和程序调试方法。

二. 仪器设备

微机，微机接口实验箱，示波器，三用表等。

三. 课设内容和要求

用查询法，将 ADC 0809 通道 0 外接 0 ~ 5V 电压，转换成数字量后，在七段 LED 数码管上，以小数点后两位（几十毫伏）的精度，显示其模拟电压的十进值；

0809 通道 0 的数字量以线性控制方式送 DAC0832 输出,当通道 0 的电压为 5V 时,0832 的 OUT 为 0V, 当通道 0 的电压为 0 时,0832 的 OUT 为 2.5V; 此模拟电压再送到 ADC 0809 通道 1, 转换后的数字量在 CRT 上以十六进制显示; 通道 0 的数字量经 74LS574 输出到八位 LED 上, 且以一定的要求, 点亮 LED 指示灯。调整电位器, 用示波器或三用表观察 0832 的变化, 观察七段 LED 数码管数值的变化, 观察 LED 灯的变化,

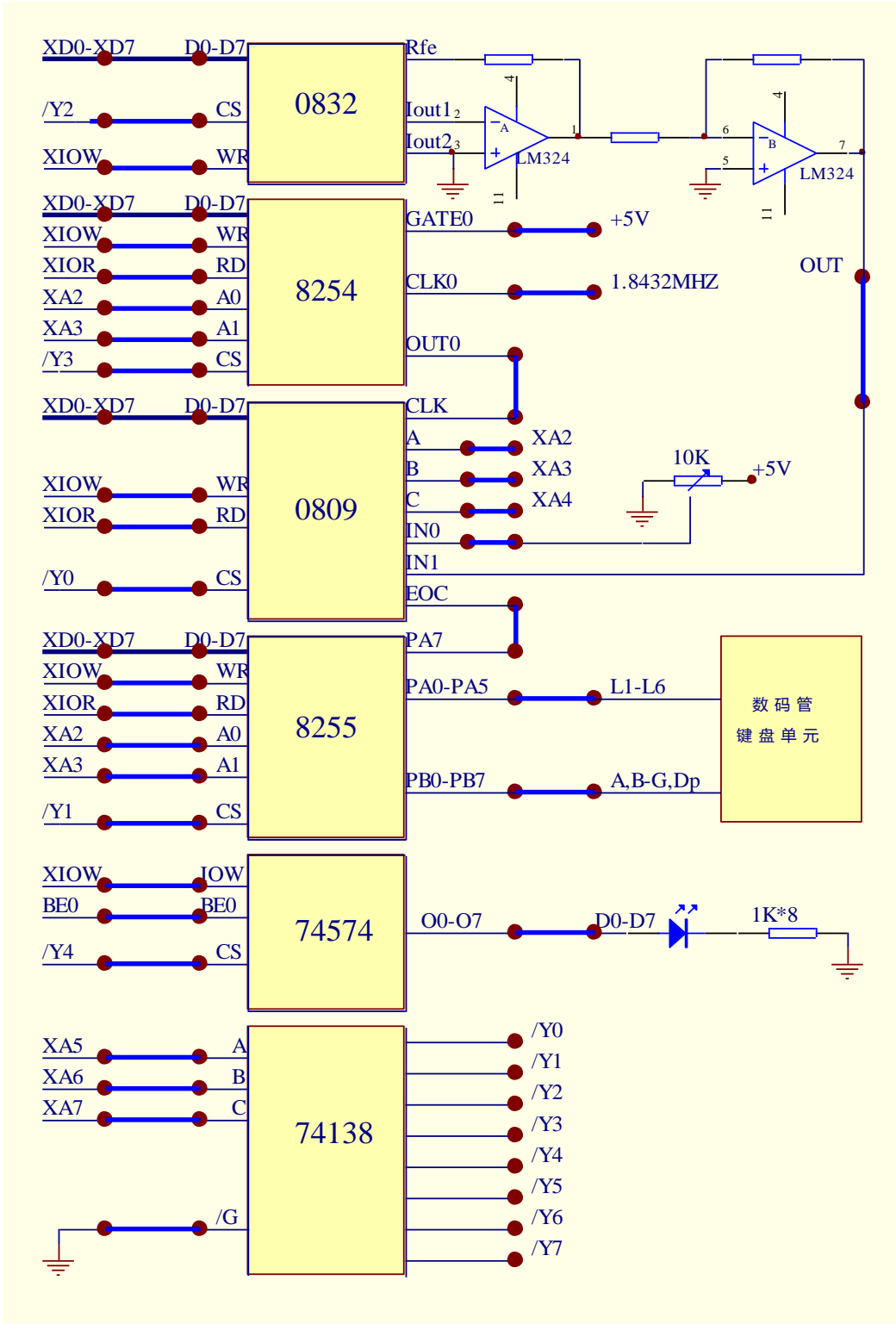
ADC 0809 的 CLK 脉冲, 由定时器 8254 的 OUT0 提供; ADC 0809 的 EOC 信号, 用 8255 的 PC0 检测; 74LS574 外接的 LED 灯变化如下: 若电压值小于 0.5V, 则最低位 (D0) LED 灯亮, 若电压值大于 4.5V, 则最高位 LED 灯亮, 若电压值在 0.5V ~ 4.5V, 则八位 LED 灯由低向高变化亮, 且高位 LED 灯亮时低位灯全亮。

四. 总体设计

- 1) ADC 0809 的 IN0 采集电位器 0 — 5V 电压, IN1 采集 0832 输出的模拟量。
- 2) DAC 0832 将 ADC 0809 的 IN0 数字量后重新转换成模拟量输出。
- 3) 8255 用于检测 ADC 0809 转换是否, 为七段 LED 数码管显示提供显示驱动信息。
- 4) 七段 LED 数码管显示 ADC 0809 的 IN0 的值。
- 5) 74LS574 驱动八位发光二极管, 使它们按要求点亮, 来指示当前采样值的范围。
- 6) 8254 提供 ADC 0809 的采样时钟脉冲。
- 7) 74LS138 译码器为各芯片提供地址信息。
- 8) 有良好的人—机对话界面。能够显示主菜单, 做功能选择。开始数据采集后, 主键盘有键按下, 退出返回 DOD 系统。

五. 硬件设计

因采用了PC机和微机实验箱，硬件电路设计相对比较简单，主要利用微机实验箱上的8255并行口、ADC 0809、DAC 0832、七段LED数码管单元、8254定时/计数器、74LS574输出接口、电位器等单元电路，就构成了数据采集系统，硬件电原理框图4-2-1所示。



4-2-1 数据采集二电原理框图

六. 软件设计

本设计通过软件编程,实现模/数转换器 0809 分别对 IN0 0-5V 直流电压的采样,并将 IN0 的值转换成十进制后,在七段数码管上显示;CPU 根据 IN0 的值,使八位发光二极管根据题目的要求,指示相应的范围,IN1 采集 0832 输出的模拟量,转换后的数字量在 CRT 上以十六进制显示。

1、设计思想

数据采集系统分成四个功能模块,分别是 A/D 转换和 D/A 转换模块、量纲转换模块、数码管显示模块、八位发光二极管驱动模块。

(1) A/D 转换和 D/A 转换模块

根据题目要求,采用查询方式实现模/数转换,在 0809 IN0 开始启动后,程序不断地检查 8255 的 PC0,直到高电平,就读取 IN0 的值并保存。此数字量分成两部分输出,一部分以线性控制的方式送给 0832 输出,再送到 ADC 0809 通道 1;另一部分将此数字量转换成十进制数后,分别送到个位、十分位、百分位存储单元保存,以供七段数码管显示时调用。

(2) 量纲转换模块

量纲转换模块简单,只要将 IN0 的数字量分别除以 51,商存入个位存储单元;余数乘以十,再除以 51,商存入十分位存储单元;余数再乘以十,除以 51,若余数大于 25,则商加 1,小于 25 则舍去,这样就达到四舍五入的精度,再把商存入百分位存储单元;以供显示时调用,量纲码转换模块可参考图 2-4 所示的流程图。

(3) 数码管显示模块

数码管显示模块比较简单,可参考实验 8255 并行口中的显示模块流程图,应注意的是在显示个位时,要加上小数点的显示,这可以在查表获取个位段码后,再加上 80H 来实现,注意每一位显示后要适当延迟时间。

(4) 八位发光二极管驱动模块

八位发光二极管驱动模块可通过某个设定的门限值,将 IN0 的数字量与其比较,确定其指示的范围,在大于 4.5V 和小于 0.5V 时,驱动处理比较简单,在 4.5V 和 0.5V 之间,需考虑如何按要求来驱动八位发光二极管。

2、流程图

数据采集二主程序流程图如图 4-2-2 所示子程序流。

七. 课程设计报告

1) 课程设计目的和内容。

- 2) 硬件设计：电原理框图及电原理框图工作过程的简要说明。
- 3) 软件设计：程序流程图和接口部件的程序段。
- 4) 编程和调试中遇到什么问题,是怎样解决的。
- 5) 运行结果，体会和建议。
- 6) 程序清单。

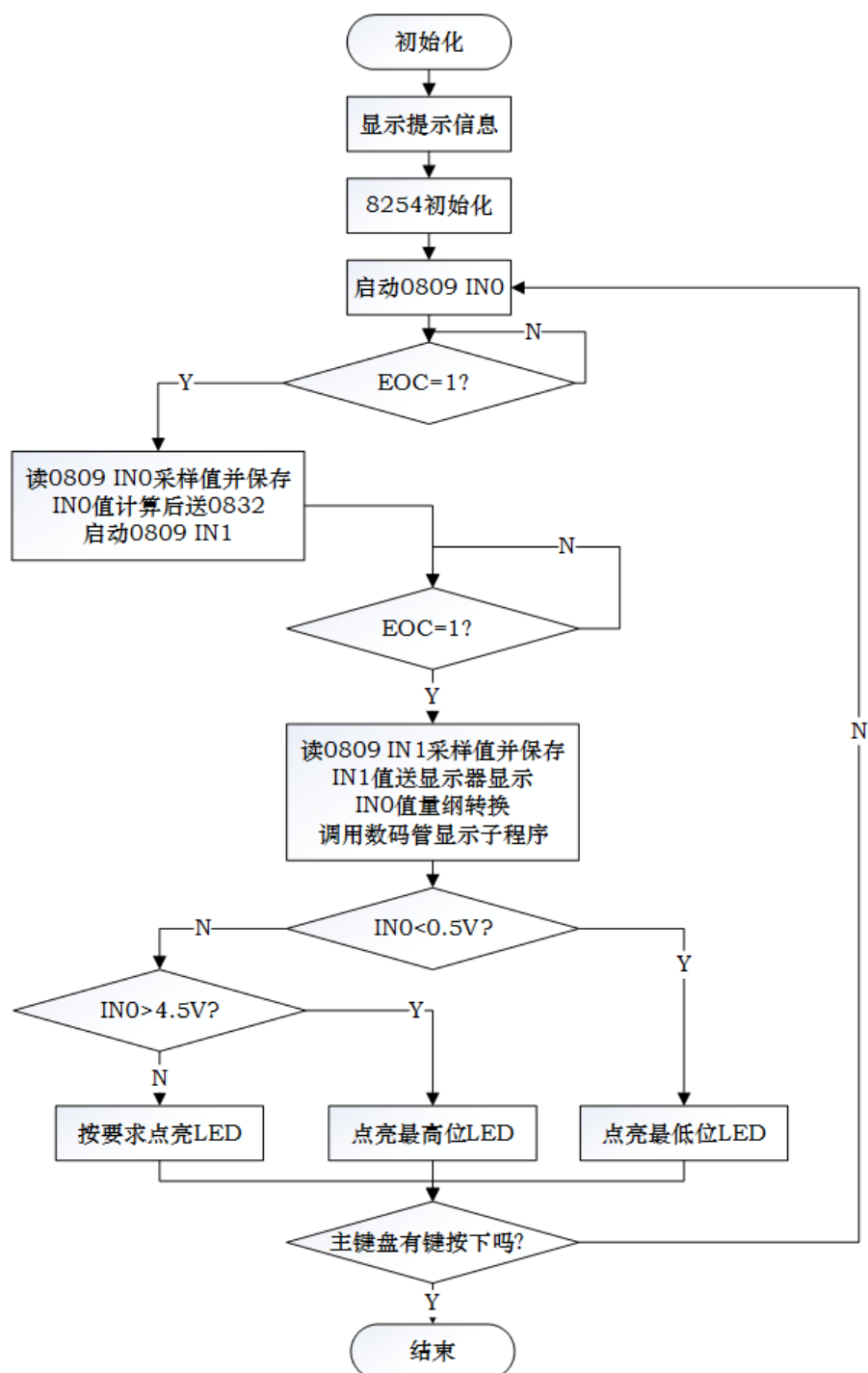


图 4-2-2 数据采集系统二主程序流程

课程设计三 数据采集系统三（中断法）

一. 课设目的

进一步掌握微机原理知识，了解微机在实时采集过程中的应用，学习、掌握编程和程序调试方法。

二. 课设内容和要求

用中断法，将 ADC 0809 通道 0 外接 0 ~ 5V 电压，转换成数字量后，在七段 LED 数码管上，以小数点后两位（几十毫伏）的精度，显示其模拟电压的十进值；

0809 通道 0 的数字量以线性控制方式送 DAC0832 输出,当通道 0 的电压为 5V 时,0832 的 OUT 为 0V, 当通道 0 的电压为 0 时,0832 的 OUT 为 2.5V; 此模拟电压再送到 ADC 0809 通道 1, 转换后的数字量在 CRT 上以十六进制显示。

ADC 0809 的 CLK 脉冲，由定时器 8254 的 OUT0 提供；ADC 0809 的 EOC 信号，用作 8259 中断请求信号。

要有较好的人机对话界面，控制程序的运行。

三. 总体设计

- 1) ADC 0809 的 IN0 采集电位器 0 — 5V 电压,IN1 采集 0832 输出的模拟量。
- 2) DAC 0832 将 ADC 0809 的 IN0 数字量后重新转换成模拟量输出。
- 3) 8259 用于检测 ADC 0809 转换是否结束和向 CPU 发送 INTR 信号
- 4) 8255 为七段 LED 数码管显示提供显示驱动信息。
- 5) 七段 LED 数码管显示 ADC 0809 的 IN0 的值。
- 6) 8254 提供 ADC 0809 的采样时钟脉冲。
- 7) 有良好的人—机对话界面。系统运行时，显示主菜单，开始数据采集，在数据采集时，主键盘有键按下，退出返回 DOD 系统。

四. 硬件设计

因采用了 PC 机和微机实验箱，硬件电路设计相对比较简单，主要利用微机实验箱上的 8255 并行口、ADC 0809、DAC 0832、七段 LED 数码管单元、8254 定时/计数器、74LS574 输出接口、电位器等单元电路，就构成了数据采集系统，硬件电原理框图 4-3-1 所示。

1、设计思想

数据采集系统分成六个功能模块，分别是主程序模块、量纲转换模块、数码管显示模块、中断服务子程序模块。

(1)主程序模块

根据题目要求，采用中断方式实现数据采集,因此可把读取 IN0 的值并保存,和此数字量经线性控制后送给 0832 输出，作为 0809 IN1 的模拟电压和 IN1 的启动、读取以及 IN0 的 BCD 码转换，放在中断服务子程序模块中完成。所以，在主程序模块上主要实现启动 0809 IN0、显示模块的调用、和中断结束部分程序。

(2) 量纲转换模块

量纲转换模块在中断服务程序内完成，只要将 IN0 的数字量分别除以 51，商存入显示 IN0 的个位存储单元；余数乘以十，再除以 51，商存入十分位存储单元；余数再乘以十，除以 51，若余数大于 25，则商加 1，小于 25 则舍去，这样就达到四舍五入的精度，再把商存入百分位存储单元；以供七段数码管显示时调用，BCD 码转换模块可参考图 10 所示的流程图。

(3) 数码管显示模块

数码管显示模块比较简单，自编程序可参考图 1-6 所示的流程图，也可参考实验 8255 并行口中的显示模块，应注意的是在显示个位时，要加上小数点的显示，这可以在查表获取个位段码后，再加上 80H 来实现,注意每一位显示后要适当延迟时间。

(4) 中断服务子程序模块

考虑到 0832 的输出是 0809 IN1 的输入,因此，为防止在执行中断服务子程序时，二次进入中断服务子程序，故在进入中断服务子程序后,应关闭中断,而在退出中断前开中断，以便下次中断进入。中断服务子程序模块主要完成对 IN0 数据的读取和保存,对 IN1 的启动和读取,对 IN0 数据的 BCD 转换等任务。

2. 参考流程图：

数据采集系统三程序流程图如图 4-3-2 所示，数码管显示子程序流程图如图 3-4-7 所示。

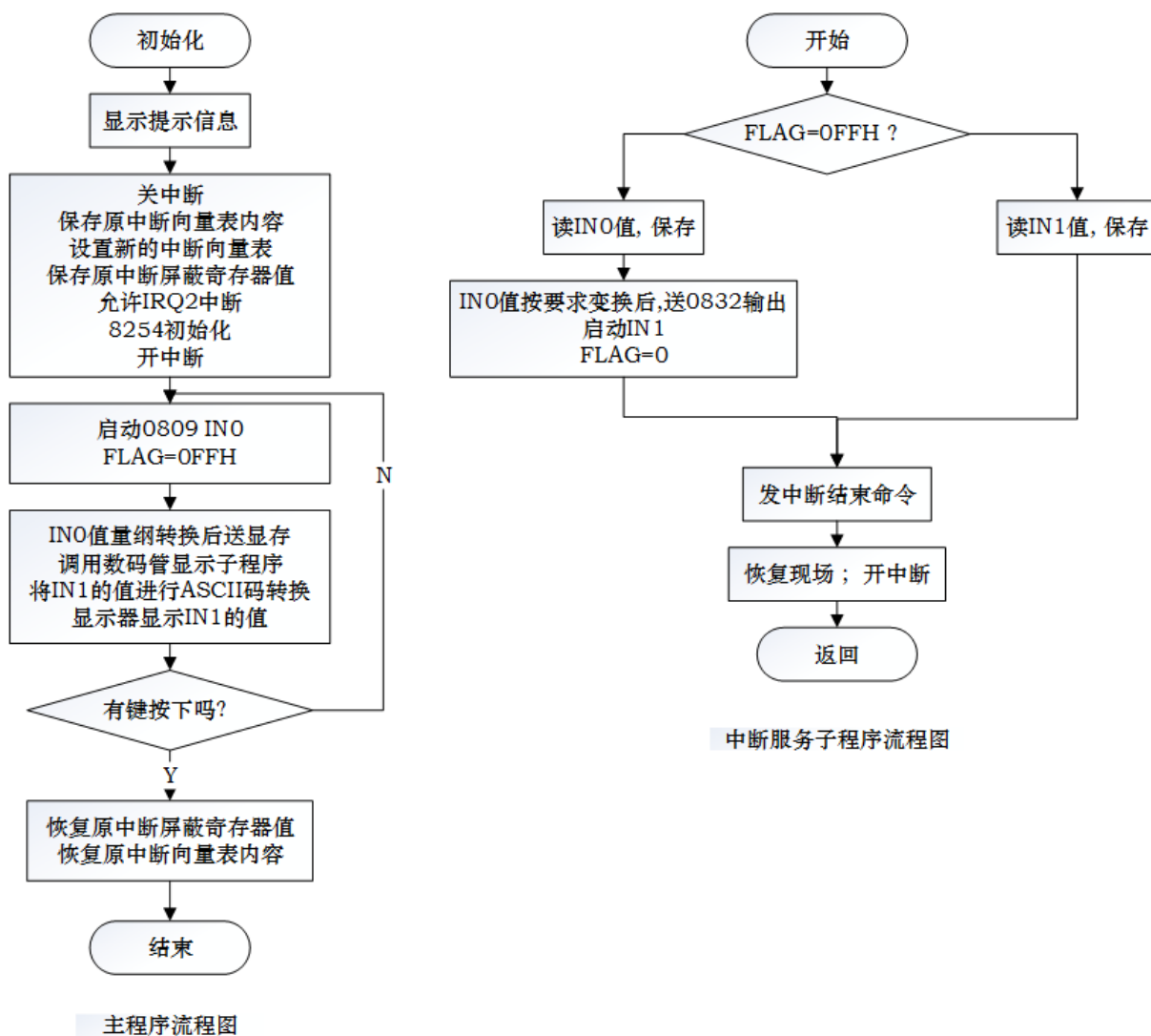


图 4-3-2 数据采集系统三程序流程图

六. 课程设计报告和内容

- 1) 课程设计目的和内容。
- 2) 硬件设计：电原理框图及电原理框图工作过程的简要说明。
- 3) 软件设计：程序流程图和接口部件的程序段。
- 4) 编程和调试中遇到什么问题,是怎样解决的。
- 5) 运行结果,体会和建议。
- 6) 程序清单。

课程设计四 信号发生器

一. 课设目的

进一步掌握微机原理知识，了解怎样用微机来实现产生各种信号的波形，学习、掌握编程和程序调试方法。进一步掌握 A/D 转换器、D/A 转换器、并行接口 8255、计数器/定时器 8254 等芯片的应用。

二. 课设内容和要求

ADC 0809 通道 0 外接 0 ~ 5V 电压，转换成数字量，在七段 LED 数码管上显示,若:

- 1) 0<数字量<33H, 则 DAC0832 输出每阶为 1V 的阶梯波。
- 2) 33H<数字量<66H, 则 DAC0832 输出正弦波。
- 3) 66H<数字量<99H, 则 DAC0832 输出梯型波。
- 4) 99H<数字量<0CCH, 则 8253 计数器 1 输出 500HZ 的方波。
- 5) 0CCH<<数字量<0FFH, 8253 计数器 1 输出 1000HZ 的方波。

要求有较好的人机对话界面；在数据采集过程中，若主键盘有键按下，则停止运行，当主键盘有键按下值是“E”时，返回 DOS。

三. 总体设计

- 1) ADC 0809 通道 0 外接电位器上的 0 ~ 5V 电压，作为控制波形发生器产生波形信息的依据。
- 2) 8254 计数器 0 为 0809 提供 CLK 时钟脉冲；计数器 1 输出 500HZ 或 1KHZ 的方波。
- 3) DAC0832 用来产生阶梯波、正弦波、梯形波。
- 4) 8255 用来检测检测 ADC 0809 的转换结束信息,同时控制七段 LED 数码管显示。
- 5) 程序开始运行时，显示主菜单,为保证 ADC 0809 数据采集的正确性，用当前采集数据 10 次的平均值，作为产生波形信息的控制依据。在程序运行中，若主键盘有键按下，停止当前所产生的波形。

四. 硬件设计

因采用了 PC 机和微机实验箱，硬件电路设计相对比较简单，主要利用微机实验箱上的 8255 并行口、ADC 0809、DAC 0832、七段 LED 数码管单元、8254 定时/计数器、电位器等单元电路，就构成了信号发生器，硬件电原理框图如图 4-4-1 所示。

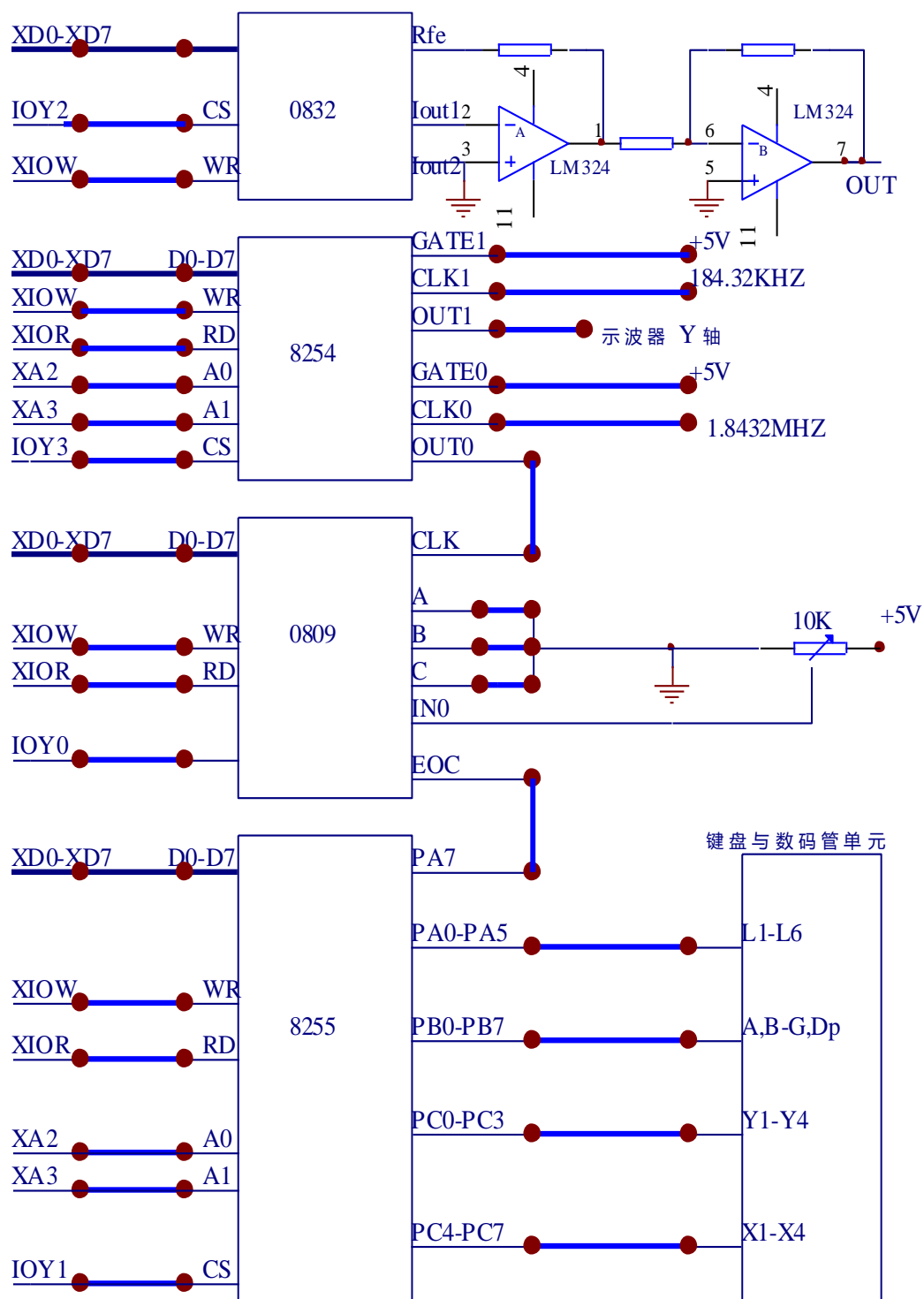


图 4-4-1 信号发生器原理框图

五. 软件设计

本设计通过软件编程，实现模/数转换器 0809 对 IN0 0-5V 直流电压的采样,在七段数码管上显示 IN0 的值;根据采样的值，CPU 将控制不同的接口单元，产生不同的信号波形。

1、设计思想

数据采集系统分成七个功能模块，分别是主程序模块、数码管显示模块、正弦波产生模块、梯形波产生模块、阶梯波产生模块、方波产生模块。

(1)主程序模块

根据题目要求，数据采集时需采集 10 次，最后取其平均值，作为结果存入存储单元，为了在数码管上显示，要调用数码管显示模块；CPU 根据采样的值，控制不同的接口单元，产生不同的信号波部形。因此，在主程序模块中，需为调用正弦波产生模块、梯形波产生模块、阶梯波产生模块、方波产生模块做准备。在程序运行中，若主键盘按下 E 键，则返回 DOS 系统；否则只是停止当前所产生的波形，重新采集 ADC 0809 通道 0 的电压，而产生相应的波形。

(2) 数码管显示模块

数码管显示模块比较简单，可参考实验 8255 并行口中的显示模块或自编。

(3) 正弦波产生模块

正弦波产生模块可先建立一个周期如表 4-4-1 所示的正弦波数值表，然后向 0832 逐个输出，就可在 0832 的 OUT1 上观察到正弦波。

表 4-4-1 正弦波数值表

80H	8CH	98H	0A5H	0B0H	0BCH	0C7H	0D1H
0DAH	0E2H	0EAH	0F0H	0F6H	0FAH	0FDH	0FFH
0FFH	0FDH	0FAH	0F6H	0F0H	0EAH	0E2H	0DAH
0D1H	0C7H	0BCH	0B0H	0A5H	098H	8CH	80H
7FH	73H	67H,	5AH	4FH	43H	38H	2EH
25H	1DH	15H	0FH	09H	05H	02H	00H
00H	02H	05H	09H	0FH,	15H,	1DH	25H
2EH	38H	43H	4FH	5AH	67H	73H	7FH

(4) 梯形波产生模块

梯形波的产生较为简单，其产生方法与锯齿波基本相同，差别就是其起始值不是 0，而是一固定的值，是 1V 或 2V 所对应的数字量。

(5) 阶梯波产生模块

阶梯波的产生可用二种方法，一种是参照正弦波的方法，另一种是在前一阶的基础上，再加上 51 或 33H，就产生了新一阶的波形，一直加到 FFH，再重复就产生了一系列的梯形波。注意：阶梯波的每阶差别是 1V。

(6) 方波产生模块

方波产生模块最简单，只要给 8253 写工作方式控制字，然后按公式：

$n = F_{clk1} / F_{out1}$ 给计数器 1 置初值,就可以产生所需的方波。式中: F_{clk1} 为计数器 1 的计数脉冲, F_{out1} 为计数器 1 的输出脉冲。

流程图

信号发生器主程序流程图如图 4-4-2 所示;阶梯波产生流程如图 4-4-3 所示; 梯形波产生流程图如图 4-4-4 所示; 正弦波产生流程如图 4-4-5 所示; 方波产生流程图如图 4-4-6 所示。

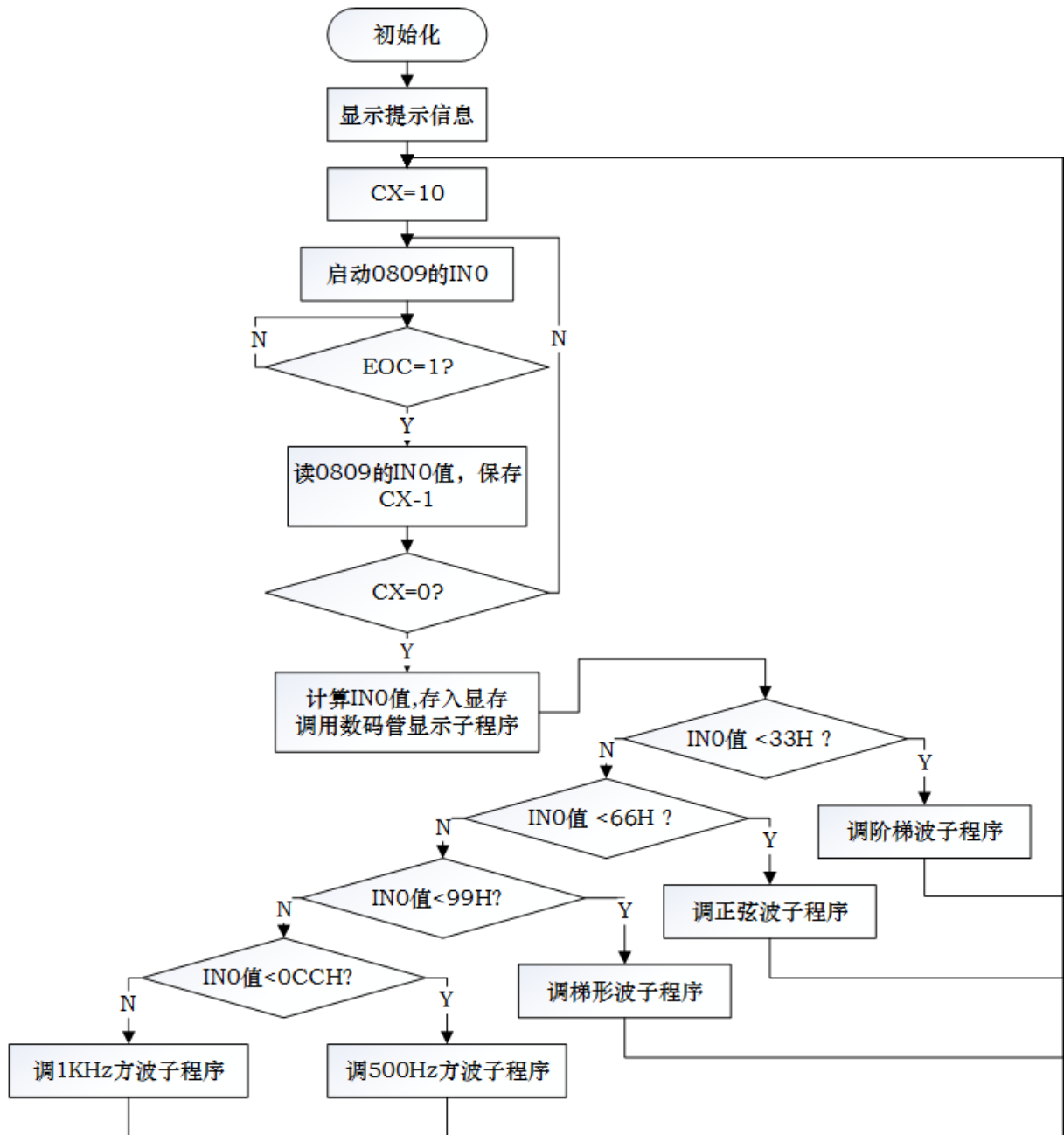


图 4--4-2 信号发生器主程序流程图

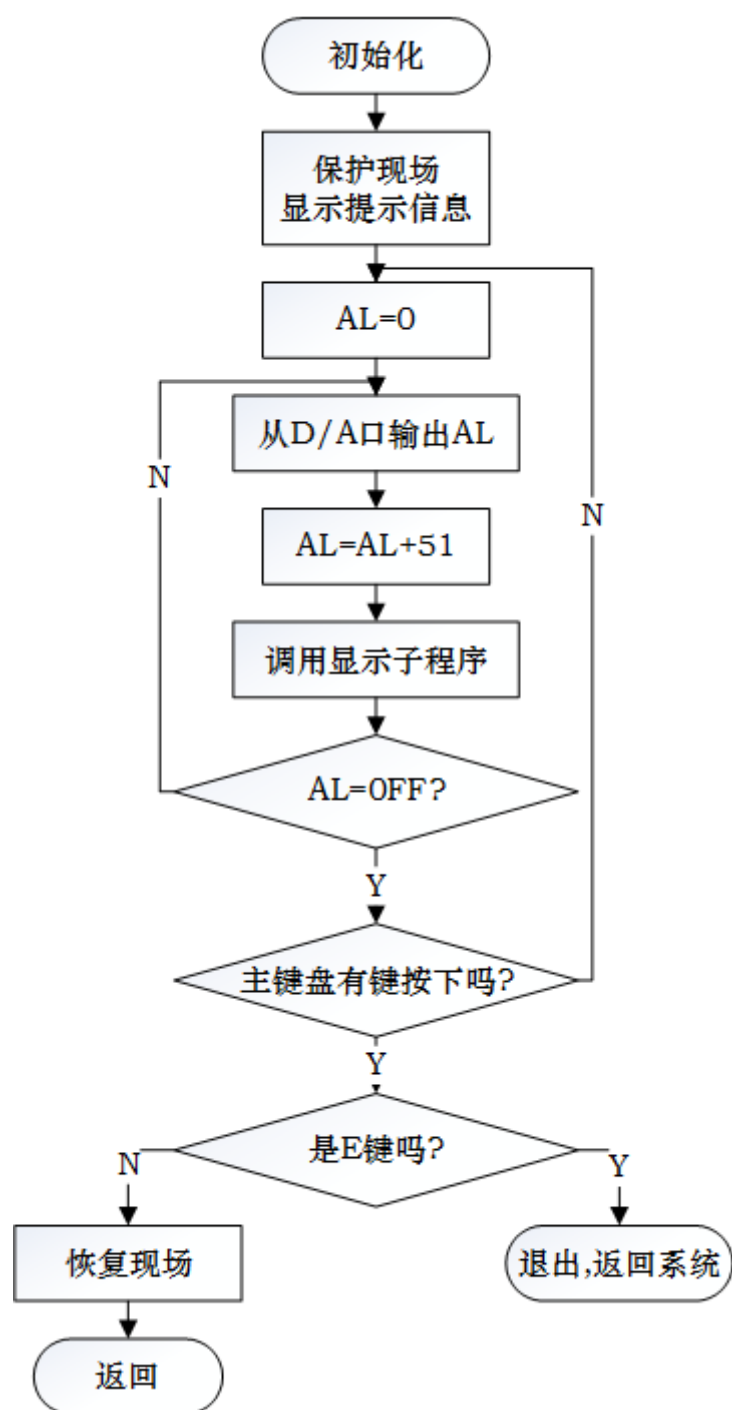


图 4—4--3 阶梯波产生流程图

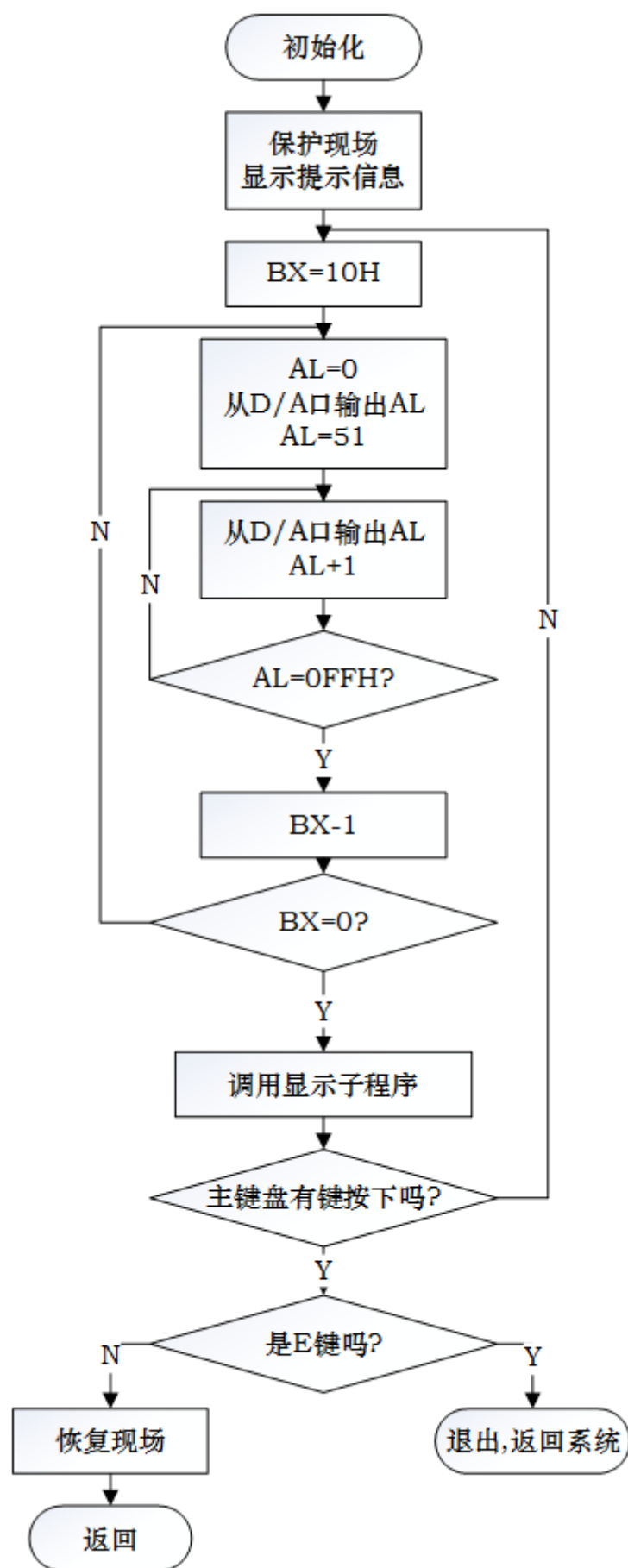


图 4-4-4 梯形波产生流程图

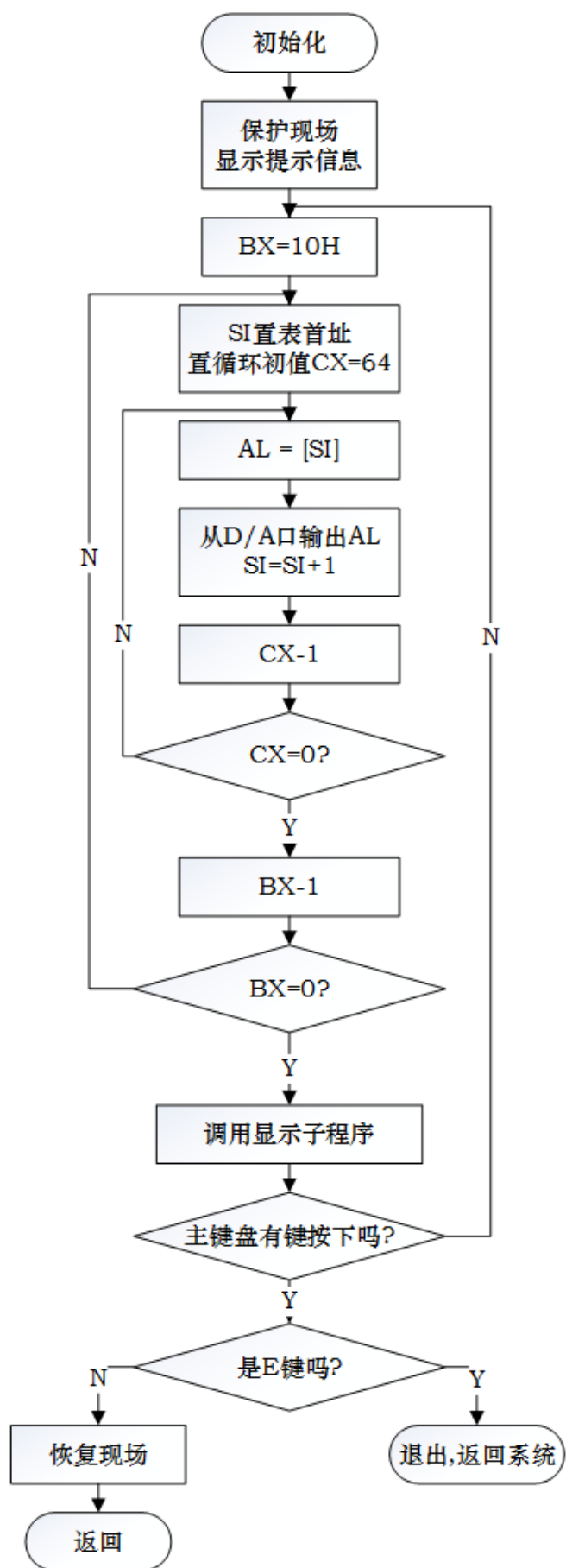


图 4-4-5 正弦波产生流程图

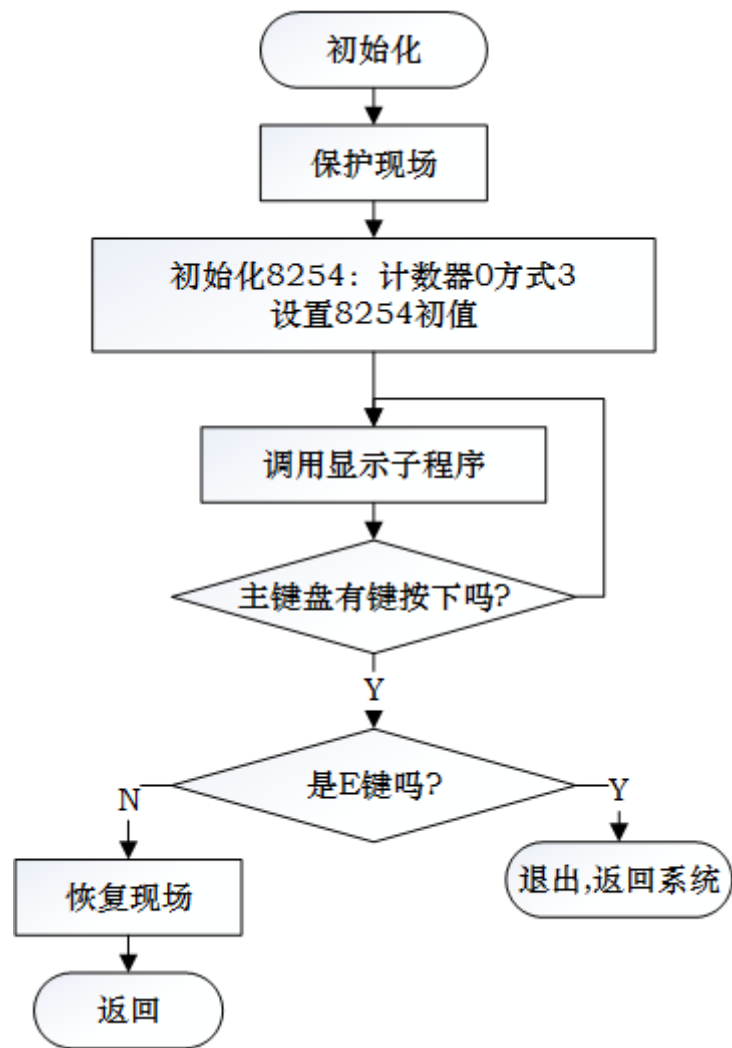


图 4-4-6 方波产生流程图

六. 课程设计报告和内容

- 1) 课程设计目的和内容。
- 2) 硬件设计：电原理框图及电原理框图工作过程的简要说明。
- 3) 软件设计：程序流程图和接口部件的程序段。
- 4) 编程和调试中遇到什么问题,是怎样解决的。
- 5) 运行结果，体会和建议。
- 6) 程序清单。

课程设计五 交通灯实时控制系统设计

一. 课设目的

进一步掌握微机原理的知识与微机在实际中的应用,掌握编程与调试方法。

二. 课设内容和要求:

利用微机实验箱,以本市现行的交通灯控制信息为依据,用查询的方法实现交通灯控制系统。要求: 通行/禁止时间在七段 LED 数码管上以倒计数的方式显示; 主干道、副干道的通行时间用主键盘来设置, 左转弯、直通、右转弯指示灯应根据通行时间来控制; 具体时间的分配与控制时间如下表 4-5-1 所示:

表 4-5-1

	主干道			副干道		
时间	左转弯灯	直通灯	右转弯灯	左转弯灯	直通灯	右转弯灯
主 $X < T < 45$	红	绿	红	红	红	红
$45 < T < 15$	红	绿	绿	红	红	绿
$15 < T < 5$	绿	红	绿	红	红	绿
$5 < T < 0$	黄	黄	黄	红	红	红
副 $X < T < 45$	红	红	红	红	绿	红
$45 < T < 15$	红	红	绿	红	绿	绿
$15 < T < 5$	红	红	绿	绿	红	绿
$5 < T < 0$	红	红	红	黄	黄	黄

注: 主 X 是主干道设定时间, 副 X 是副干道设定时间。

三. 总体设计

- 1) 8255 的 A 口, B 口外接七段 LED 数码管, 用来实现通行/禁止时间的显示。C 口外接键盘, 一用于设定主干道, 副干道通行时间。
- 2) 8254 作为定时器, 定时时间为一秒。
- 3) 74LS245 和 74LS574 口控制主干道左转弯、直通、右转弯指示灯; B 口控制副干道左转弯、直通、右转弯指示灯。
- 4) 采用共阴极的双色发光二极管作为指示灯, 当红灯管脚为高电平, 绿灯管脚为低电平时, 指示灯为红色; 当红灯管脚为低电平, 绿灯管脚为高电平时, 指示灯为绿色; 当红灯管脚绿灯管脚同为高电平时, 指示灯为黄色。
- 5) 有良好的人—机对话界面。程序运行时, 首先显示主菜单, 在提示信息下分别输入干道/副干道通行时间, 运行后当主键盘有键按下时, 退出运行。

四. 硬件设计

因采用了 PC 机和微机实验箱，硬件电路设计相对比较简单，利用微机实验箱上的 8255 并行口、8254 计数/定时器、七段 LED 数码管单元，再加上共阴极的双色发光二极管，便构成了交通灯控制系统，硬件电原理框图如图 4-5-1 所示。

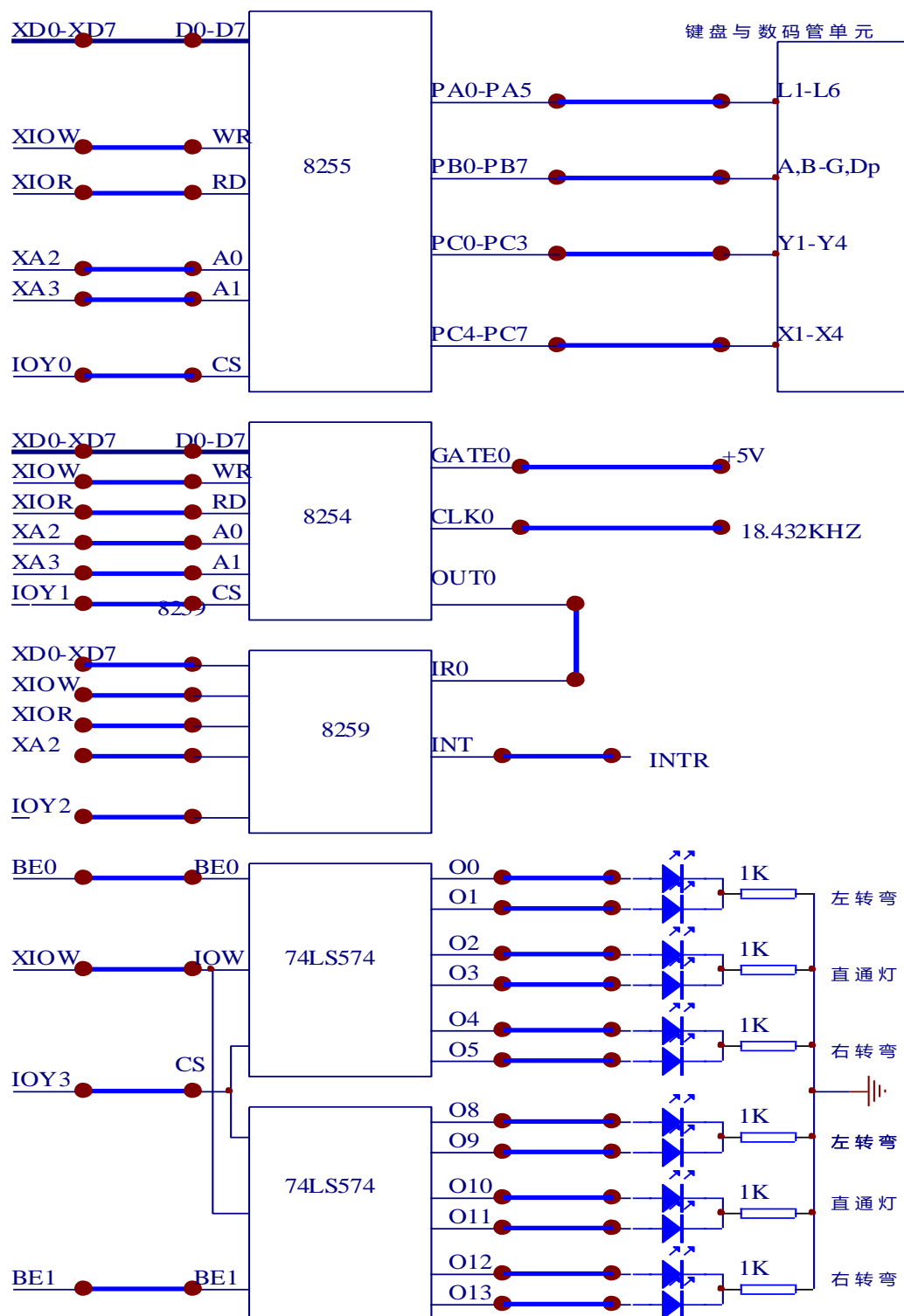


图 4-5-1 交通灯控制系统电原理框图

五. 软件设计

本设计通过软件编程,使 8254 输出定时信号,当 8255 的 C 口测出其为高电平时,将原计时器的内容减 1,比较当前计时器的值是大于 35 秒,还是在 35 秒和 15 秒之间,或者在 15 秒和 5 秒之间,或在 5 秒之内,CPU 根据计时器的内容,发出不同的控制信息给 8255,从而控制主、副干道的左转弯、直通、右转弯指示灯,实现对交通灯的控制。

1、设计思想

控制系统分成四个功能模块,分别是定时器模块、数码管显示模块、通行模块、时间计算模块。

(1) 定时器模块

定时器模块相对比较简单,它可在主程序中完成,只要给 8254 写工作方式控制字和设置初值,就可以实现每秒产生一个脉冲的信号,初值 $n = F_{clk0} / F_{out0}$,式中 F_{clk0} 为 $clk0$ 脉冲的频率, F_{out0} 为 F_{out0} 输出频率。

(2) 数码管显示模块

数码管显示模块也比较简单,可参考实验 8255 并行口中的显示模块,实现主、副干道的通行/禁止时间的显示,区别是在此的段码、位码由 I/O 口控制。

(3) 通行模块

通行模块根据计时器的值,和通行标志,发出相应的控制信息,实现主、副干道的通行/禁止的控制。

(4) 时间计算模块

时间计算模块可分成二个部分,首先实现对计时器的值减 1,检查计时器的值是否为 0,若是 0,在改变通行标志的同时给计时器赋不同的初值;若不是 0,不改变通行标志和计时器的值;然后将计时器的值进行 BCD 码转换,十位和个位的值分别送到 LED 显存,以供显示模块显示。

2、程序流程图

交通灯控制系统主程序流程图如图 4-5-2 所示;通行子程序流程图如图 4-5-3 所示;时间计算子程序流程图如图 4-5-4 所示。

六. 课程设计报告和内容

- 1) 课程设计目的和内容。
- 2) 硬件设计:电原理框图及电原理框图工作过程的简要说明。
- 3) 软件设计:程序流程图和接口部件的程序段。
- 4) 编程和调试中遇到什么问题,是怎样解决的。

- 5) 运行结果，体会和建议。
- 6) 程序清单。

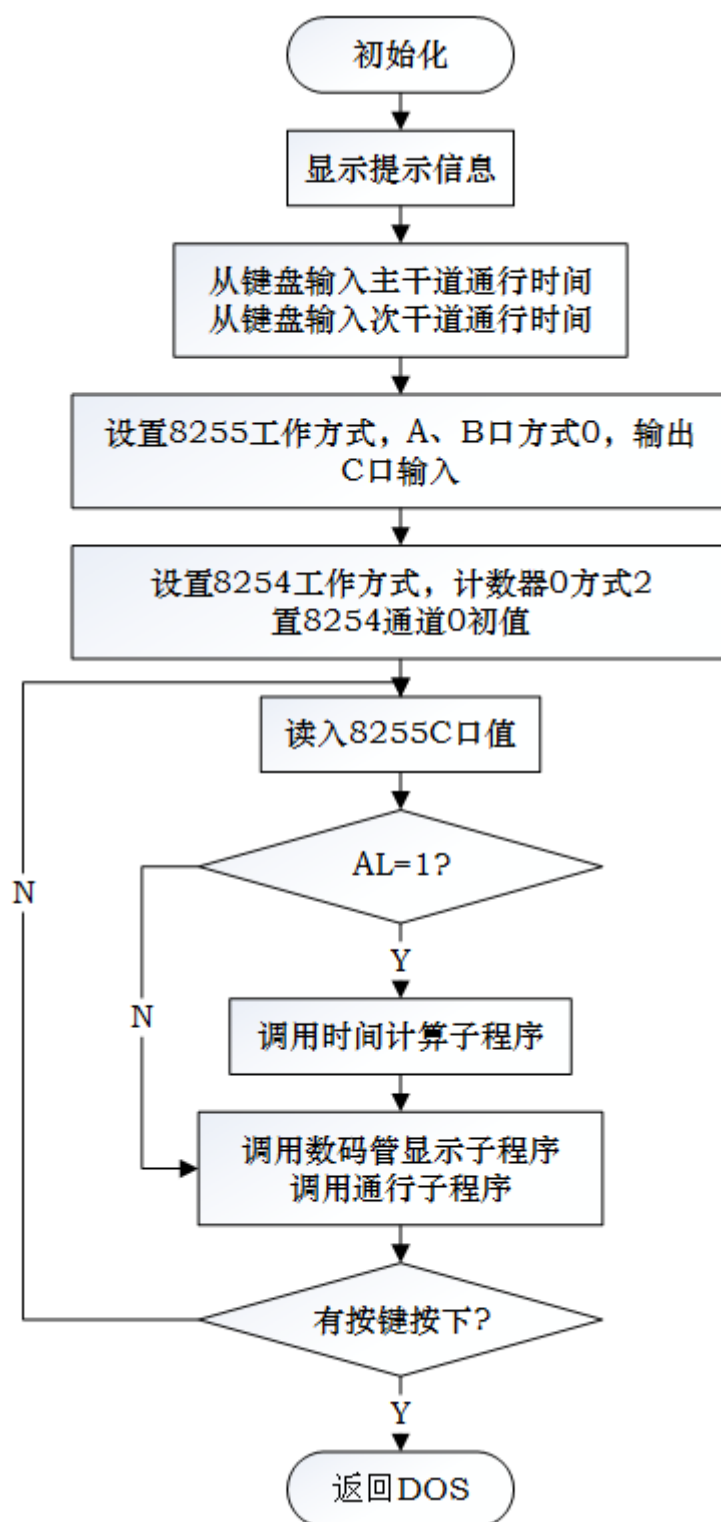


图 4-5-2 交通灯控制系统主程序流程图

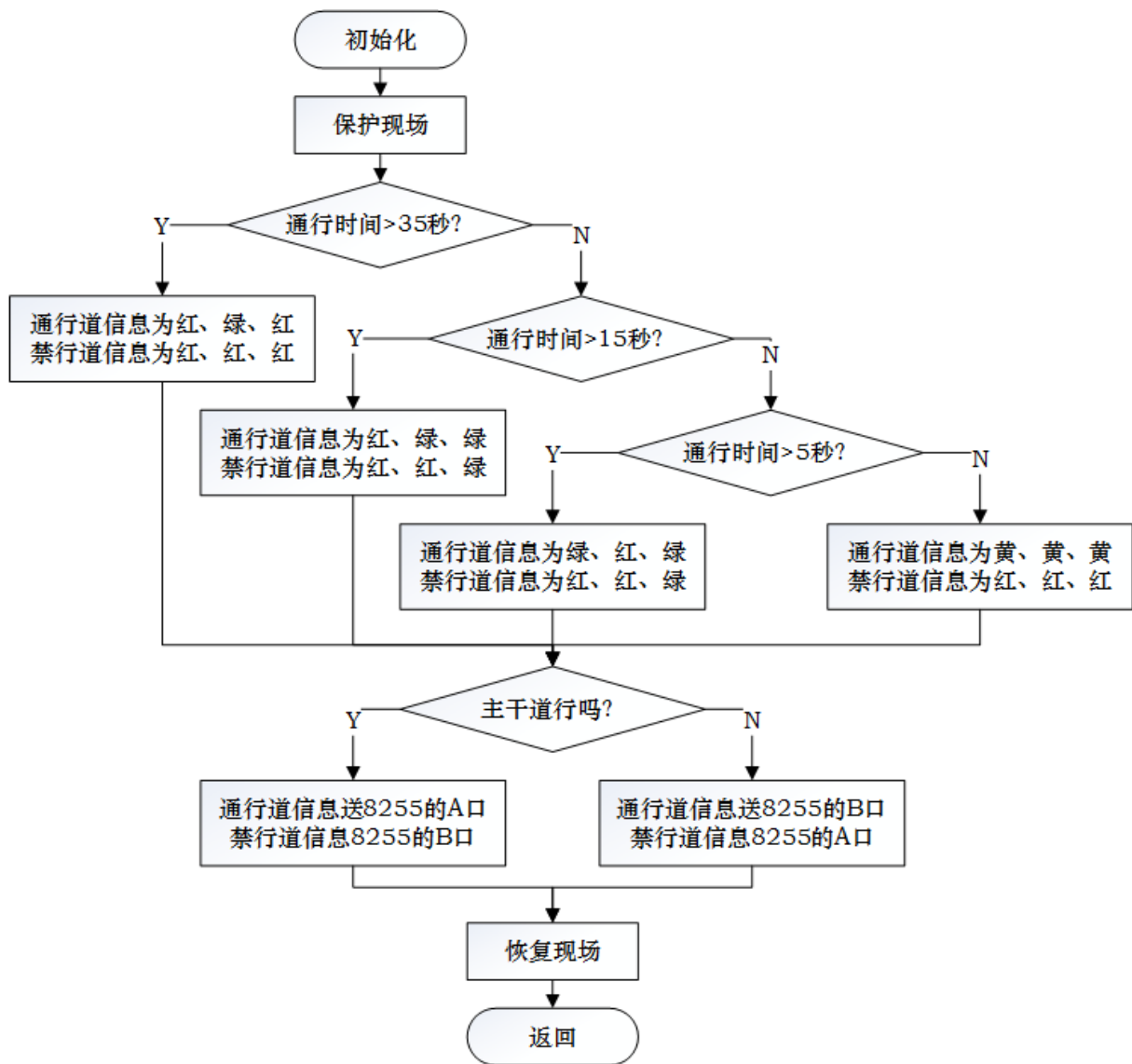


图 4-5-3 通行子程序流程图

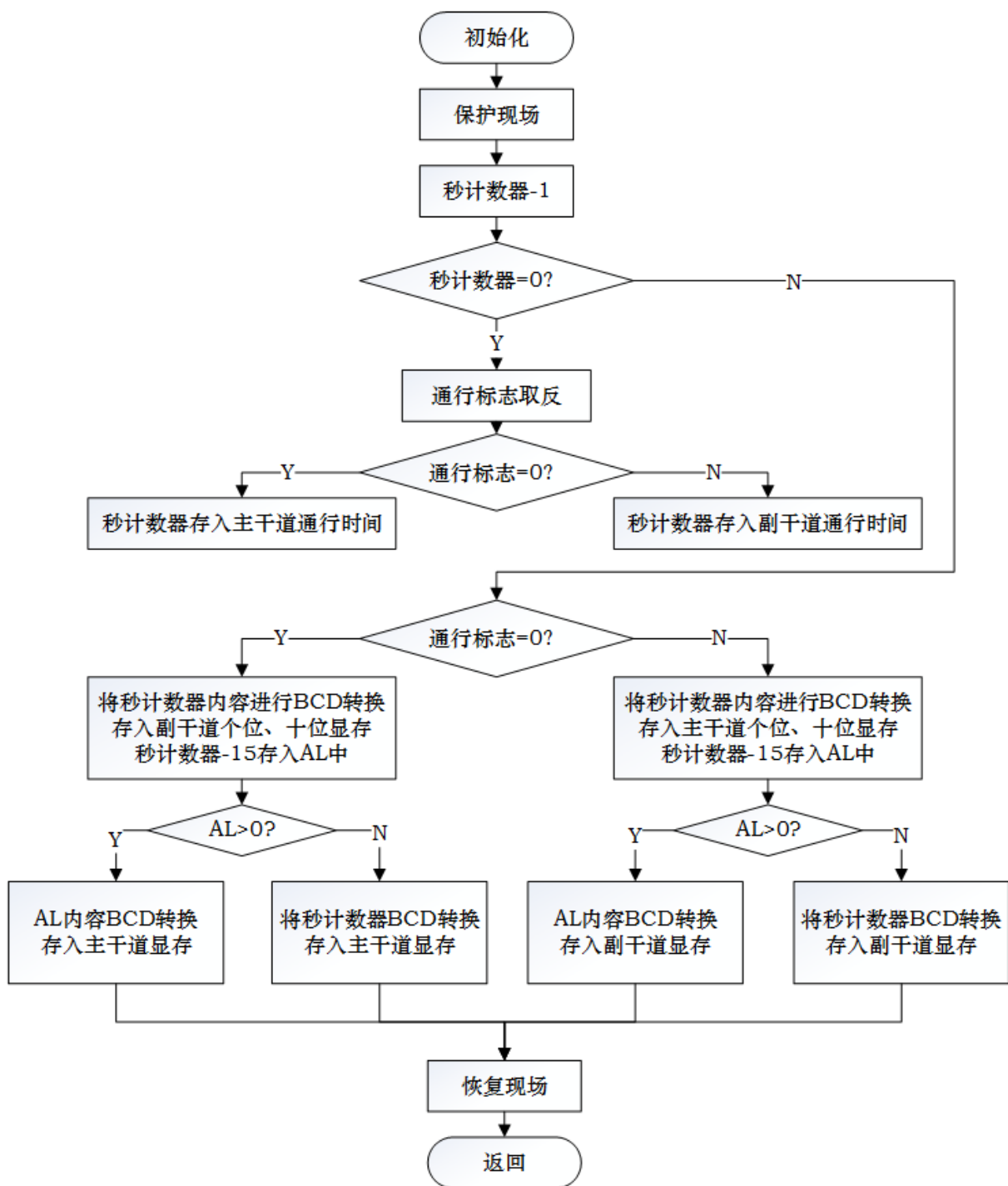


图 4-5-4 时间计算子程序流程图

课程设计六 步进电机控制系统设计

一. 课设目的

进一步掌握微机原理知识,了解怎样用微机来实现步进电机转速的控制,学习、掌握编程和程序调试方法。进一步掌握并行接口 8255、计数器/定时器 8254 等芯片的应用。

二. 课设内容

用中断方法实现步进电机转速和方向的控制,转动方向和每秒转速在七段数码管上显示。要求的转速为 1 次/秒,3 次/秒,5 次/秒,15 次/秒,75 次/秒五档,转动方向为顺时针和逆时针。有良好的人—机对话界面。

程序运行时,首先显示主菜单,在提示信息下从主键盘完成运行速度和方向的设置。步进电机顺时针转为 L,逆时针转为 R;程序运行后当主键盘有键按下时,退出运行。

三. 总体设计

- 1) 8255 的 A 口输出步进电机控制信息,经驱动电路后驱动步进电机转动
- 2) 8254 作为定时器,定时时间根据要求来设定。
- 3) 74LS245 和 74LS574 外接七段 LED 数码管,用来实现的步进电机转动方向和转速显示。

四. 硬件设计

(一)、四相步进电机工作原理

步进电机是机电一体化的关键部件之一,被广泛应用需要精确定位同步行程控制等场合。所谓步进,就是指每给步进电机一个递进脉冲,步进电机各绕组的通电顺序就改变一次,即电机转动一次。根据步进电机控制绕组的多少可以将电机分为三相、四相和五相。实验平台可连接的步进电机为四项八拍电机,电压为 DC12V,其励磁线圈及其励磁顺序如图 4-6-1 及表 4-6-1 所示。

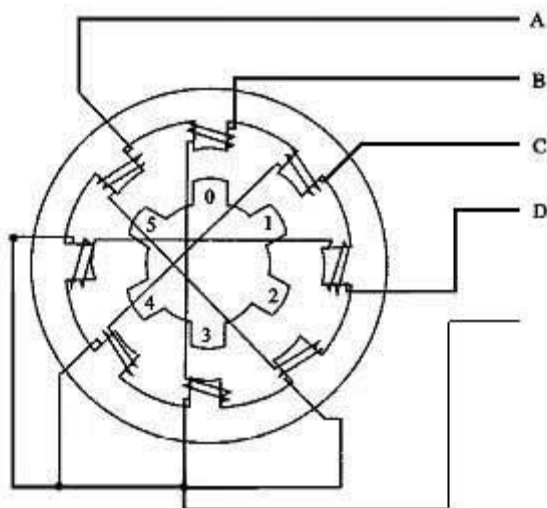


图 4-6-1 四相五线步进电机

表4-6-1 励磁顺序

步序	D	C	B	A
1	0	0	0	1
2	0	0	1	1
3	0	0	1	0
4	0	1	1	0
5	0	1	0	0
6	1	1	0	0
7	1	0	0	0
8	1	0	0	1

除四相绕组 ABCD 外，还有一个公共抽头，用于提供电机的工作电压。以四相八拍的运行方式为例，其顺时针转动的励磁顺序为(逆时针恰好相反):A→AB→B→BC→C→CD→D→DA→A，见表 4-6-1。

通电一周的周期越短,即驱动频率越高，则电机转速越快，但步进电机的转速也不可能太快，因为它每走一步需要一定的时间，若信号频率过高，可能造成电机失步,甚至只在原地颤动。

(二)电原理框图

因采用了 PC 机和 PC 总线接口应用平台,硬件电路相对简单,硬件电原理框图如图 4-6-2 所示。图中 ULN2803 是正与非 OC 驱动器;8254 的作用是输出信号中断请求信号,8259 的作用向 CPU 输出信号中断请求信号, 8255 输出步进电机走步的控制信号,及进电机转动方向和转速显示功能。

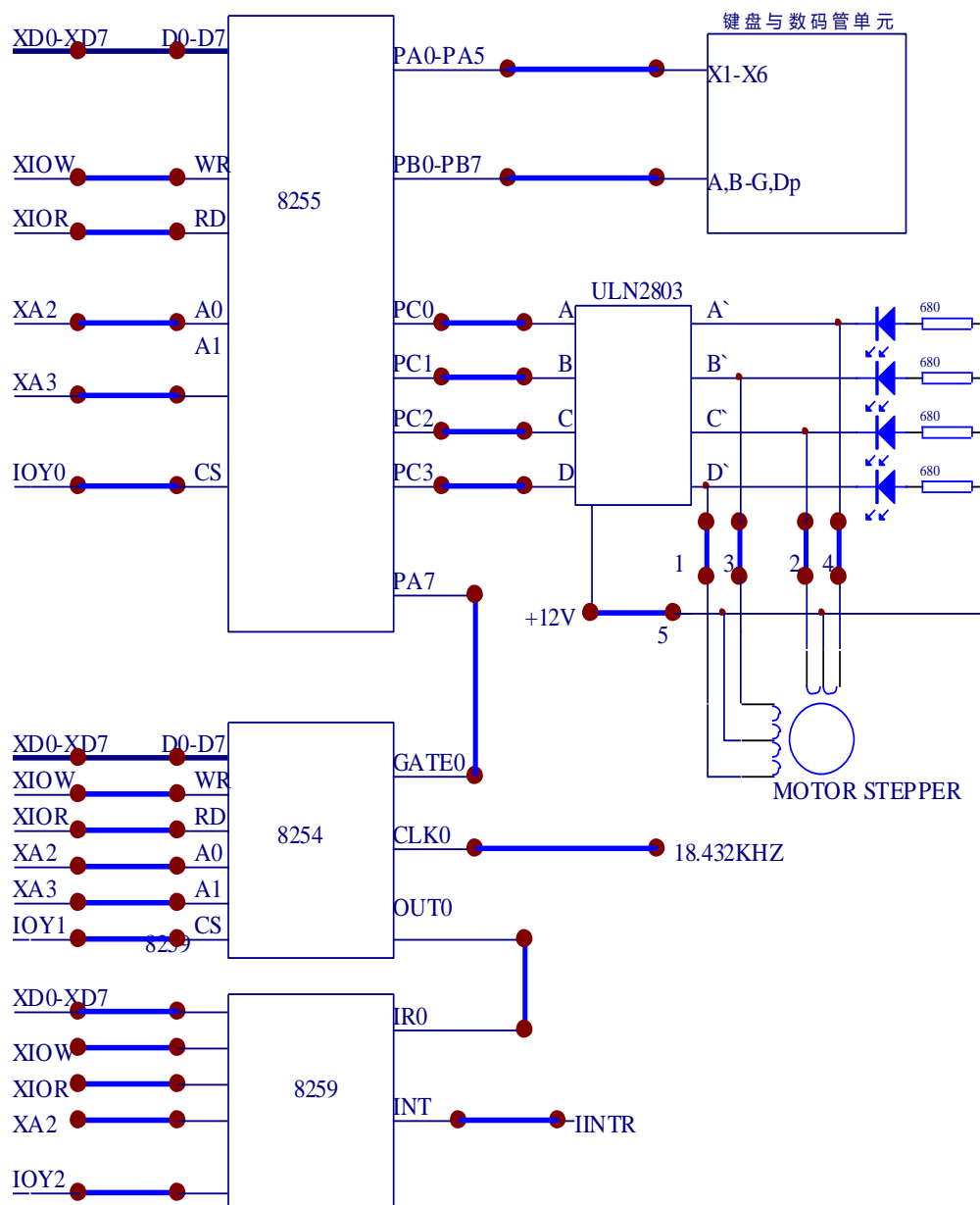


图 4-6-2 步进电机控制系电原理框图

五. 软件设计

本设计通过软件编程使 8254 输出定时信号产生申请中断,当 CPU 发出命令由 8255 的 A 口输出脉宽信号来控制步进电机的走步。电机的转动和停止,是通过 8255 的 B7 端子输出高、低电平,使 8254 的计数是继续还是暂停来控制中断申请。8254 的定时时间决定了电机转速的快慢。

控制系统分六个功能模块,分别是主程序,转速设计,转向设计,键盘扫描,数码管显示,中断服务子程序模块,在前两个功能模块中都设计了 ESC 键来取消或暂停执行当前操作。

1. 转速设置

由于采用中断方法实现控制步进电机转速,每次进入中断服务程序,控制步进电机就转动一次,因此,控制进入中断的频率,就控制步进电机转速,而中断申请是由 8254 的计数器来实现的,赋给

8254 不同的计数初值,就有不同的电机转速。步进电机的转速和 8254 的计数初值如表 4-6-2 所示。

表 4-6-2

步速	计数初值
1	18432
3	6144
5	3686
15	1228
75	245

2.转向设置

在内存单元中设置一方向标志 flag1,假设 flag1=0 时,电机为顺时针转动,而当 flag1=1 时,电机为逆时针转动,判断操作在中断子程序中进行。

3.中断子程序

在中断子程序中,将预先设计步进电机内存单元的值,循循环左移或右移一位,通过 8255 的 C 输出,控制步进电机的相序变化,从而使电机连续的转动,左移或右移将使电机顺时针或逆时针转动,它由 FLAG1 来决定.

4. 键盘扫描和数码管显示

键盘扫描和描数码管显示可参看 8255 实验。

6. 程序流程图

程序流程框图如图 4-6-3、图 4-6-4 所示。

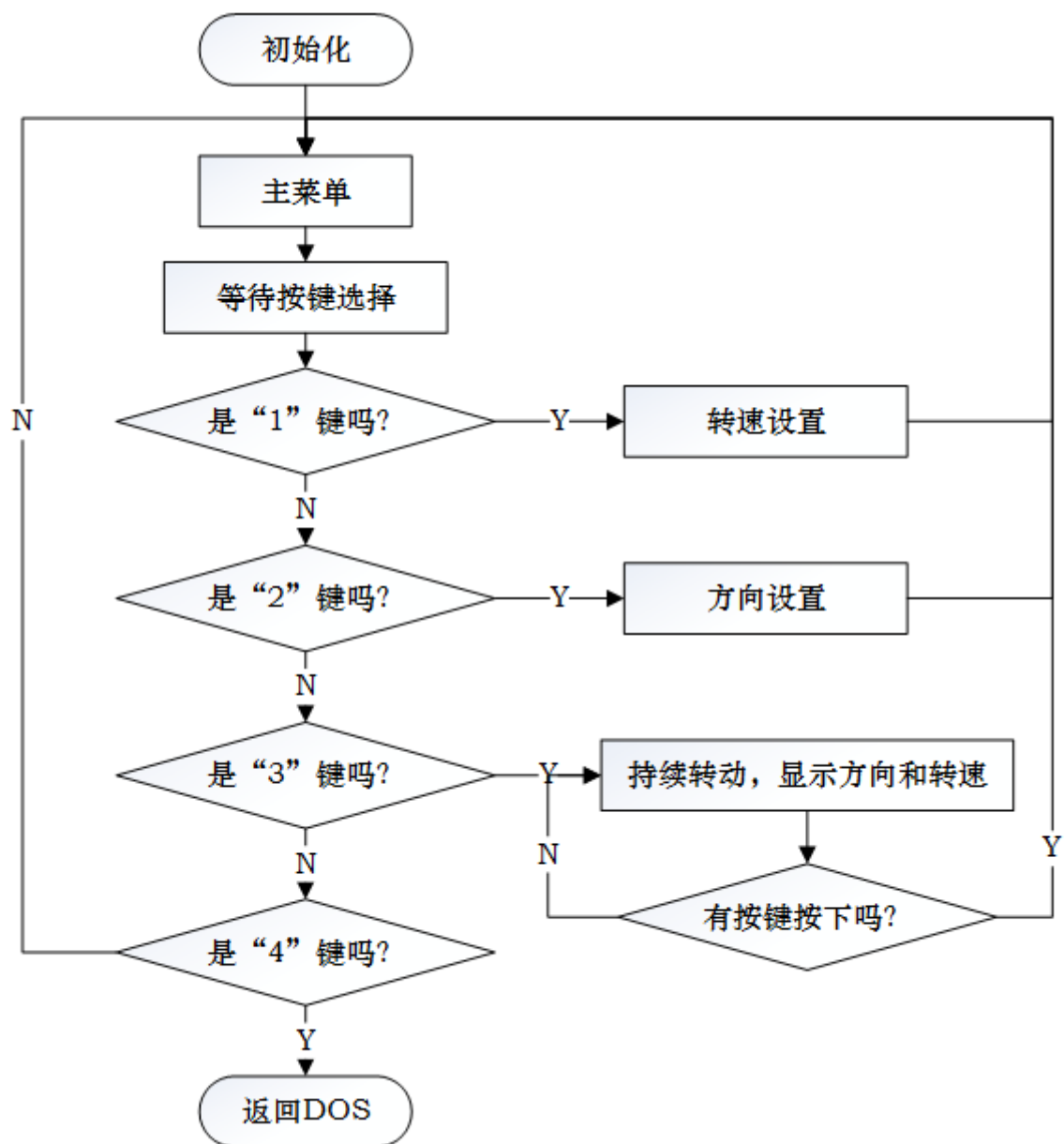


图 4-6-3 步进电机控制系统主程序流程图

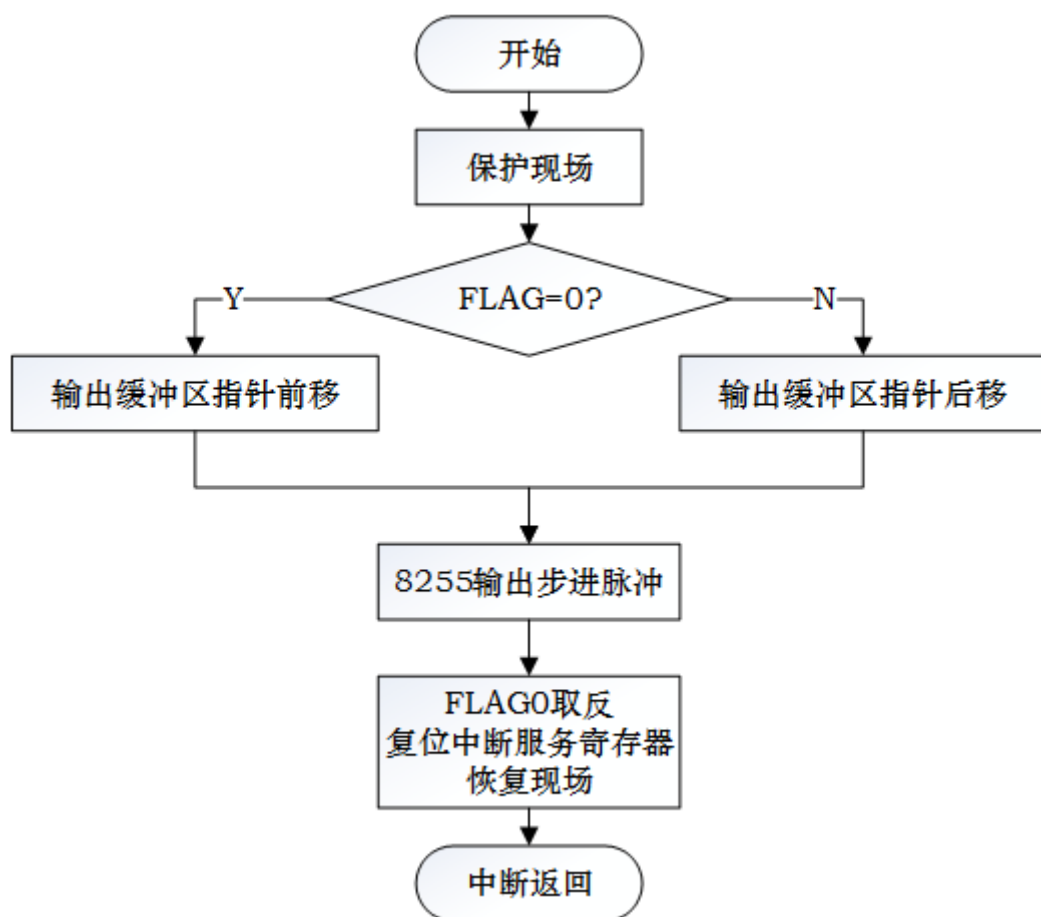


图 4-6-4 步进电机控制系统中断服务子程序流程图

七. 课程设计报告和内容

- 1) 课程设计目的和内容。
- 2) 硬件设计：电原理框图及电原理框图工作过程的简要说明。
- 3) 软件设计：程序流程图和接口部件的程序段。
- 4) 编程和调试中遇到什么问题,是怎样解决的。
- 5) 运行结果，体会和建议。
- 6) 程序清单。

附录 A 常用 DOS 系统功能 (INT 21H)

功能号	功 能	入 口 参 数	出 口 参 数
00H	程序结束	CS=程序段前缀	无
01H	键盘输入并回显		(AL)=输入字符
02H	显示输出	DL=输出字符	
03H	异步通讯输入		AL=输入数据
04H	异步通讯输出	DL=输出数据	
05H	打印机输出	DL=输出字符	
06H	I/O 直接控制台	DL=FF(输入) DL=字符(输出)	AL=输入字符
07H	键盘输入(无回显)		AL=输入字符
08H	键盘输入(无回显)		AL=输入字符
09H	显示字符串	DS:DX=串首地址 '\$'结束字符串	
0AH	键盘输入到缓冲区	DS:DX=缓冲区首地址 (DS:DX)=缓冲区最大字符数 (DS:DX+1)=实际输入的字符数	
0BH	检验键盘状态		AL=00 有输入 AL=FF 无输入
0CH	清除输入缓冲区并 请求指定输入功能	AL=输入功能号 (1,6,7,8,A)	
25H	设置中断向量	DS:DX=中断向量 AL=中断类型号	
2AH	取日期		CX:DX=日期 CX=年 DH=月 DL=日 AL=星期几
2BH	置日期	CX:DX=日期 (同 2AH 出口参数)	若 AL=00 成功 AL=0FF 失败
2CH	取时间		CX:DX=时间 CH=小时 CL=分 DH=秒 DL=百分秒
2DH	置时间	CX:DX=日期 (同 2CH 出口参数)	若 AL=00 成功 AL=0FF 失败
35H	取中断向量	ES:BX=中断向量 AL=中断类型	

附录 B Tdpit 集成操作软件使用说明

1. 软件运行环境

操作系统: Windows2000/XP

CPU: 奔腾 300MHz 以上

内存: 64MB 以上

显示器: 标准 VGA

硬盘: 50MB 以上

软件安装完成后默认的目录结构如下。

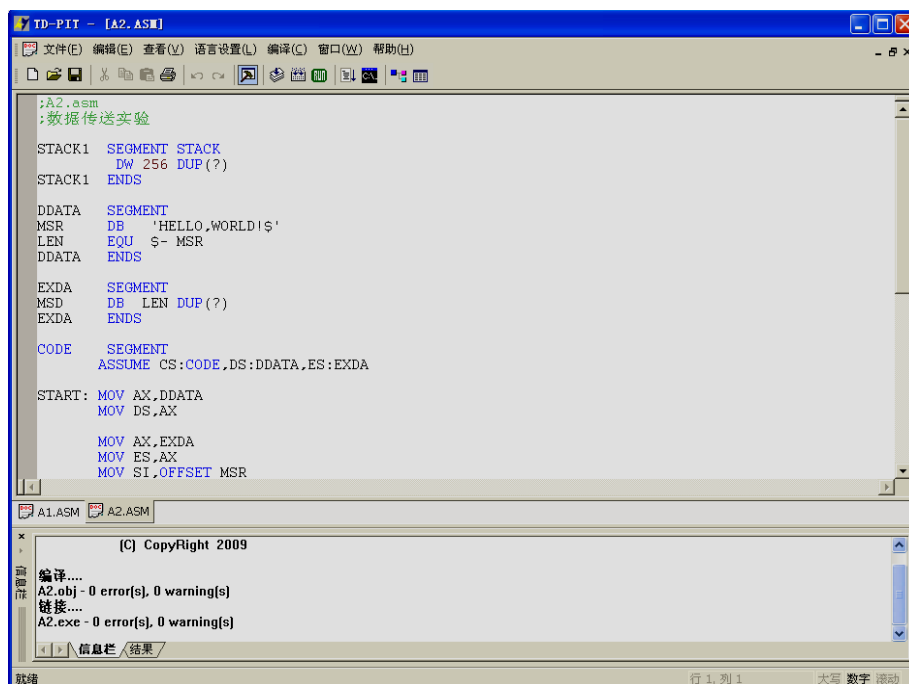
C:\Tangdu\PITD\Asm 汇编参考实验程序

C:\Tangdu\PITD\C C 语言参考实验程序

2. 软件概述

Tdpit 集成操作软件是在 Windows2000/XP 系列操作系统下进行汇编和 C 语言接口实验的集成编辑调试环境。它允许用户在该环境下编辑、编译、运行及源语言级调试汇编和 C 语言程序。并且提供了详细的软件操作及实验操作帮助系统,用户可以在实验过程中随时查看实验操作步骤、方法等帮助说明。该环境还集成大量的 Windows 接口应用实验例程,使用户可以很方便地完成 Windows 下接口应用实验。

软件运行后的主界面如附图 B-1 所示。主要分为两部分: 程序编辑区和信息栏。



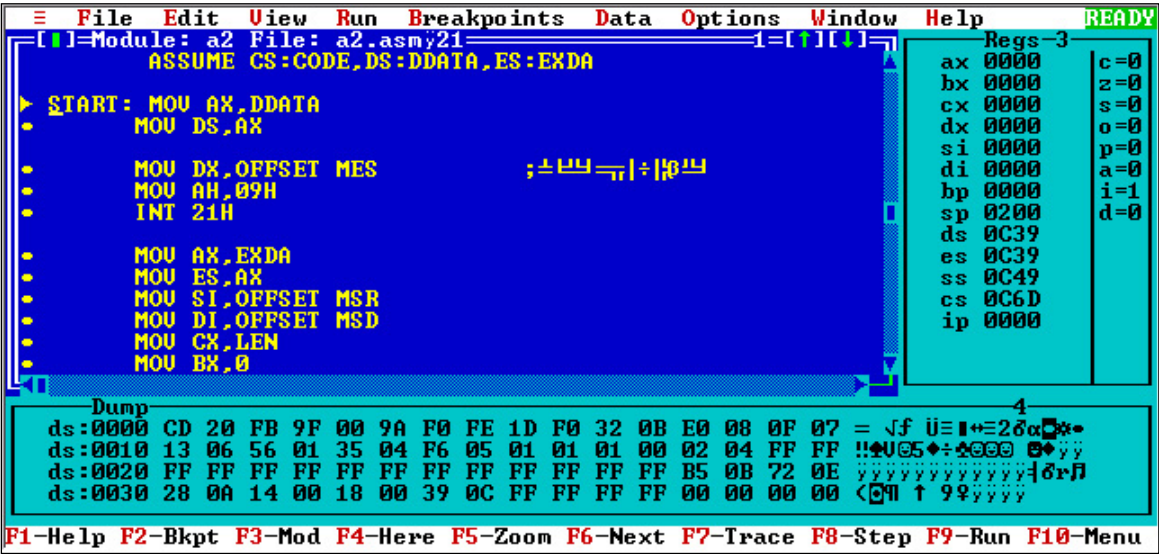
附图 B-1 Tdpit 集成操作软件运行界面

(1) 程序编辑区

位于界面上部区域，用户可在程序编辑区用“新建”命令打开一个新文档或用“打开”命令打开一个已存在的文档，在文档中用户可编辑程序。用户可在程序编辑区打开多个文档，点击文档标签可激活任一文档。编译、链接、加载以及调试命令只针对当前活动文档。用户调试程序时，将切换到调试界面中，调试界面如附图 B-2 所示。

(2) 信息栏

位于界面下部，主要显示编译和链接的结果，如果编译时有错误或警告，双击错误或警告信息，错误标识符会指示到相应的有错误或警告的行。



附图 B-2 调试界面窗口

3. 软件操作说明

1. 菜单功能介绍

(1) 文件菜单项

文件菜单如附图 B-3 所示，提供了以下命令。

新建 (N)：用此命令在 Tdtpit 中建立一个新文档。

打开 (O)：用此命令在窗口中打开一个现存的文档。您可同时打开多个文档，点击某文档的标签可激活此文档。您可用窗口菜单在多个打开的文档中切换。

关闭 (C)：用此命令来关闭当前活动文档。Tdtpit 会建议您在关闭文档之前保存对您的文档所做的改动。如果您没有保存而关闭了一个文档，您将会失去自从您最后一次保存以来所做的所有改动。在关闭一无标题的文档之前，Tdtpit 会显示另存为对话框，建议您命名和保存文档。

保存 (S)：用此命令将当前活动文档保存到它的当前的文件名和目录下。当您第一次保存文档

时, Tdpit 显示另存为对话框以便您命名您的文档。如果在保存之前, 您想改变当前文档的文件名和目录, 您可选用另存为命令。

另存为 (A) …: 用此命令来保存并命名活动文档。Tdpit 会显示另存为对话框以便您命名您的文档。



附图 B-3 文件菜单项

打印 (P) …: 用此命令来打印一个文档。在此命令提供的打印对话框中, 您可以指明要打印的页数范围、副本数、目标打印机, 以及其它打印机设置选项。

打印预览 (V): 用此命令按要打印的格式显示活动文档。当您选择此命令时, 主窗口就会被一个打印预览窗口所取代。这个窗口可以按它们被打印时的格式显示一页或两页。打印预览工具栏提供选项使您可选择一次查看一页或两页, 在文档中前后移动, 放大和缩小页面, 以及开始一个打印作业。

打印设置 (R) …: 用此命令来选择一台打印机和一个打印机连接。在此命令提供的打印设置对话框中, 您可以指定打印机及其连接。

最近使用文件: 您可以通过此列表, 直接打开最近打开过的文件, 共四个。

退出 (X): 用此命令来结束您 Tdpit 的运行阶段。您也可使用在应用程序控制菜单上的关闭命令。Tdpit 会提示您保存尚未保存的改动。

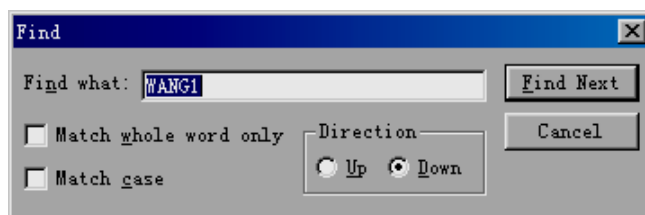
(2) 编辑菜单项

编辑菜单如附图 B-4 所示, 提供了以下命令。



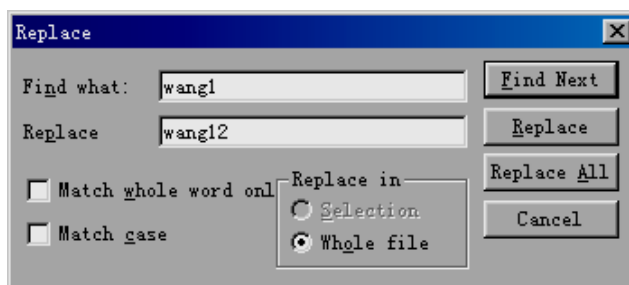
附图 B-4 编辑菜单项

- 撤销：可用此命令来撤销上一步操作。如果无法撤销上一步操作，菜单的撤销命令会变灰。
- 重复：可用此命令来恢复撤销的编辑操作。如果无法恢复撤销的编辑操作，菜单上的重复命令会变灰。
- 剪切 (T)：用此命令将当前被选取的数据从文档中删除并放置于剪贴板上。如当前没有数据被选取时，此命令则不可用。把数据剪切到剪贴板上将取代原先存放在那里的内容。
- 复制 (C)：用此命令将被选取的数据复制到剪贴板上。如当前无数据被选取时，此命令则不可用。把数据复制到剪贴板上将取代以前存在那里的内容。
- 粘贴 (P)：用此命令将剪贴板上内容的一个副本插入到插入点处。如剪贴板是空的，此命令则不可用。
- 查找：点击此命令将弹出查找对话框，如附图 B-5 所示，用于查找指定字符串。
- “Find what:” 编辑框：写入你想要查找的字符串。
- “Match whole word only” 复选框：是否全字匹配。如果不选中此复选框，找到的字符串的长度有可能大于想要查找的字符串，如：我们想要查找字符串 ‘WANG1’，可能会找到字符串 ‘WANG10’，这是因为我们没有选中全字匹配复选框。
- “Match case” 复选框：是否辨认大小写。如果不选中此复选框，找到的字符串中字符的大小写可能与我们想要查找的字符串有差别，如：我们想要查找字符串 “WANG1”，可能会找到字符串 “Wang1”。
- “Up” 单选按钮：从下向上查找
- “Down” 单选按钮：从上向下查找
- “Find Next” 按钮：查找下一个字符串，如果是第一次查找则从当前光标处开始向下或向上开始查找，如果不是第一次查找，则从上一次找到的位置向下或向上开始查找。
- “Cancel” 按钮：取消查找对话框。



附图 B-5 查找对话框

- 替换：点击此命令将弹出替换对话框，如附图 B-6 所示，找到某一字符串，并用指定字符串替换之。

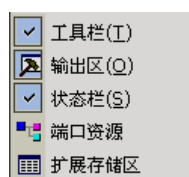


附图 B-6 替换对话框

- “Replace” 编辑框：替换后的字符串。
- “Selection” 单选按钮：如果文档中有选中部分，此按钮使能，选中此按钮则从选中部分查找和替换。
- “Whole file” 单选按钮：从整个文档中查找和替换。
- “Find Next” 按钮：查找下一个字符串。如果是第一次查找，从当前光标位置开始查找，如果不是第一次查找，则从上一次找到的位置开始查找。
- “Replace” 按钮：替换一个字符串。如果当前已经找到某一字符串，用指定字符串替换它，并找到下一个字符串，如果还没有找到某一字符串，不进行替换并找到字符串。
- “Replace All” 按钮：用指定字符串替换全部能够找到的字符串。
- “Cancel” 按钮：取消替换对话框。

(3) 查看菜单项

查看菜单如附图 B-7 所示，提供了以下命令。



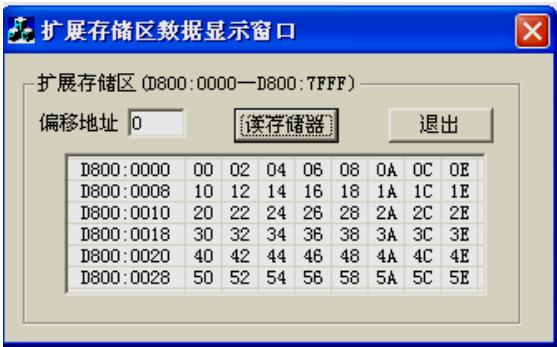
附图 B-7 查看菜单项

- 工具栏 (T): 用此命令可显示和隐藏工具栏。工具栏包括了 Tdptit 中一些主要命令的按钮, 如文件打开。在工具栏被显示时, 一个打勾记号出现在该菜单项目的旁边。
- 输出区 (O): 用此命令可显示和隐藏输出区 (信息栏)。
- 状态栏 (S): 此命令可用来显示和隐藏状态栏。状态栏描述了被选取的菜单项目或被按下的工具栏按钮, 以及键盘的锁定状态将要执行的操作。当状态栏被显示时, 在菜单项目的旁边会出现一个打勾记号。
- 端口资源: 此命令可用来查看实验系统被分配的端口资源, 在系统总线上共有 4 个 I/O 片选 IOY0~IOY3, 每个 I/O 空间所对应的地址范围在弹出的窗口中给出。如附图 B-8 所示。



附图 B-8 查看端口资源

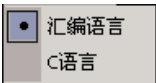
- 扩展存储区: 此命令可用来查看并修改 PITD 上 SRAM 单元中的存储器中的内容。如图 B-9 所示。



附图 B-9 扩展存储区

(4) 语言设置菜单项

语言设置菜单如附图 B-10 所示, 提供了以下命令。



附图 B-10 语言设置菜单项

汇编语言: 此命令用来设置软件的当前语言环境为汇编语言环境, 此时可以编辑、编译、链接

和调试汇编语言程序。

C 语言：此命令用来设置软件的当前语言环境为 C 语言环境，此时可以编辑、编译、链接和调试 C 语言程序。

(5) 编译菜单项

编译菜单如附图 B-11 所示，提供了以下命令。



附图 B-11 编译菜单项

编译 (C)：编译当前活动文档中的源程序，在源文件目录下生成目标文件。如果有错误或警告生成，则在输出区显示错误或警告信息，双击错误或警告信息，可定位到有错误或警告的行，修改有错误或警告的行后应重新“编译”。编译时自动保存源文件中所做的修改。

链接 (L)：链接编译生成的目标文件，在源文件目录下生成可执行文件。如果有错误或警告生成，则在输出区显示错误或警告信息，查看错误或警告信息修改源程序，修改后应重新“编译”和“链接”。

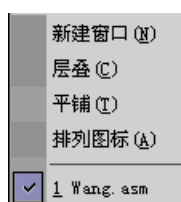
运行 (R)：执行当前连接成功的可执行程序。当前激活的程序编译连接成功或者该程序已经编译过，可执行程序已经存在，这时就可运行该程序。如果该程序没有连接成功，或者没有被连接过，可执行程序不存在，则不可以执行“运行”。所有实验例程均设计为按任意键退出运行状态。

调试 (D)：打开调试环境进行当前程序的调试。每次打开或者新建一个新的程序，都必须先进行编译连接，然后才可以执行该操作，进入调试环境。调试完毕后按“Alt + X”键退出调试环境。调试环境的操作详见调试环境的帮助。

显示模式：可以将程序调试或者运行环境调整到全屏或者窗口模式。

(6) 窗口菜单项

窗口菜单如附图 B-12 所示，提供了以下命令。



附图 B-12 窗口菜单项

新建窗口 (N)：用此命令来打开一个具有与活动的窗口相同内容的新窗口。您可同时打开数个文档窗口以显示文档的不同部分或视图。如果您对一个窗口的内容做了改动，所有其它包含同一文档的窗口也会反映出这些改动。

层叠 (C)：用此命令按相互重叠形式来安排多个打开的窗口。

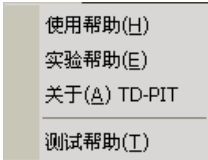
平铺 (T)：用此命令按互不重叠形式来安排多个打开的窗口。

排列图标 (A)：用此命令在主窗口的底部安排被最小化的窗口的图标。如果在主窗口的底部有一个打开的窗口，则有可能会看不见某些或全部图标，因为它们在这个文档窗口的下面。

窗口选择：Tdpit 在窗口菜单的底部显示出当前打开的文档窗口的清单。有一个打勾记号出现在活动的窗口的文档名前。从该清单中挑选一个文档可使其窗口成为活动窗口。

(8) 帮助菜单项

帮助菜单如附图 B-13 所示，提供了以下命令。



附图 B-13 帮助菜单项

使用帮助 (H)：用此命令来显示帮助的开场屏幕。从此开场屏幕，您可跳到关于使用 Tdpit 的一些指令以及各种不同类型参考资料。

实验帮助 (E)：用此命令来显示帮助的开场屏幕。从此开场屏幕，您可跳到关于使用 Tdpit 系列实验系统做实验时的一些相关信息。其中为每个实验提供了详细的实验说明及相关的实验原理，参考设计流程及实验步骤和实验接线等。

关于 (A) TD-PIT ：用此命令来显示您的 Tdpit 版本的版权通告和版本号码。

测试帮助 (T)：用此命令来显示基于 PCI 桥接卡扩展到设备上的 PC 机资源分配是否正确。


2. 工具栏功能介绍

(1) 标准工具栏

标准工具栏共有十个按钮，如附图 B-14 所示。



附图 B-14 标准工具栏

 新建按钮：用此按钮在 Tdpit 中建立一个新文档。



打开按钮：用此命令在一个新的窗口中打开一个现存的文档。



保存按钮：用此命令来关闭当前活动文档。



剪切按钮：用此命令将当前被选取的数据从文档中删除并放置于剪贴板上。



复制按钮：用此命令将被选取的数据复制到剪贴板上。



粘贴按钮：用此命令将剪贴板上内容的一个副本插入到插入点处。



打印按钮：用此命令来打印一个文档。



撤消按钮：如果可能的话，可用此命令来撤消上一步编辑操作。



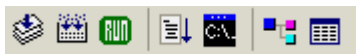
重复按钮：如果可能的话，可用此命令来恢复撤消的编辑操作。



显隐输出区按钮：用此按钮可显示和隐藏输出区。

(2) 编译工具栏

编译工具栏共有七个按钮，如附图 B-15 所示。



附图 B-15 编译工具栏



编译按钮：编译当前活动文档中的源程序，在源文件目录下生成目标文件。



链接按钮：链接编译生成的目标文件，在源文件目录下生成可执行文件。



运行按钮：执行当前连接成功的可执行程序。



调试按钮：打开调试环境进行当前程序的调试。



进入 DOS 环境按钮：此按钮提供一个进入 DOS 环境的快捷工具，若想进入 DOS 环境进行命令行操作，可以按此按钮。



查看端口资源按钮：此按钮功能可用来查看实验系统被分配的端口资源。



扩展存储区显示按钮：此按钮可以查看并修改当前 PITD 上的 SRAM 单元中的存储器内容。

4. 编程信息

当实验系统通电时，实验系统自动向 PC 机申请配置资源。但打开 Td-Pit 软件后，Td-Pit 把

Windows 的资源进行了重新配置，把 I/O、中断线都设置成固定值。如表 B-1 所示。

附表 B-1 端口分配范围

IOY0	3000H~303FH
IOY1	3040H~307FH
IOY2	3080H~30BFH
IOY3	30C0~30FFH
INTR	IRQ10

为了在 Windows 下可以完成存储器实验，Td-Pit 把存储空间 MY0 映射到了 D8000H~DFFFFH。

实验程序中资源设置举例

```

;*****

IOY0      EQU      3000H      ;片选 IOY0 对应的端口始地址

IOY1      EQU      3040H      ;片选 IOY1 对应的端口始地址

IOY2      EQU      3080H      ;片选 IOY2 对应的端口始地址

IOY3      EQU      30C0H      ;片选 IOY3 对应的端口始地址

INTR_IVADD EQU      01C8H      ;INTR 对应的中断矢量地址

INTR_OCW1 EQU      0A1H       ;INTR 对应 PC 机内部 8259 的 OCW1 地址

INTR_OCW2 EQU      0A0H       ;INTR 对应 PC 机内部 8259 的 OCW2 地址

INTR_IM    EQU      0FBH      ;INTR 对应的中断屏蔽字

;*****
```