微型计算机  
实验报告

组号： 39

### 1. 实验原理

1.1 存储器（6264）

6264为8K×8位的静态RAM电路，它为28线的双列直插式器件，其中A0～A12为13条地址线，I/O0～I/O7为8条数据线，CS2CS1为两个片选端，OE为数据输出选通端，WR为写信号端，其工作方式选择如下表所示。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 控制信号工作方式 | CS1 | CS2 | OE | WR | 数据线 |
| 读 | L | H | L | H | 输出 |
| 写 | L | H | X | L | 输入 |
| 非选 | H | X | X | X | 高阻态 |
| 非选 | X | L | X | X | 高阻态 |
| 输出禁止 | L | H | H | H | 高阻态 |

1.2 可编程键盘与显示器接口（8279）

8279在编程后对键盘，显示器进行扫描，记下按下键的坐标并显示数据。

8279最多可外接8×8的键盘及16×8的发光二极管显示器。

8279与CPU之间的连线有：

D0～D7 数据线

IRQ 中断申请信号，高电平有效

RD，WR 读写信号

RESET 复位信号

CS 片选信号

A0 缓冲器地址。为0表示数据口，为1表示命令，状态口

CLK 外部时钟输入，用以产生内部100KHz的信号，输入信号频率低于2MHz

8279与外部的连线有：

SL0～SL7 扫描信号，用于键盘与显示器的扫面。

RL0～RL7 键输入信号，有内部电阻拉成高电平。

SHIFT,CNTL/STB 键控制线，有内部电阻拉成高电平。

OUTB0～3，OUTA0～3 输出显示数据。

BD 消隐信号。

1.3 模数转换及其接口（ADC0809）

ADC0809为一个带8路模拟输入的8位逐次比较的A/D转换器，在电路内，有三态输出锁存器，其转换时间为100微妙。

它是一个28线的双列直插式器件，其引线包括：

DB0～DB7 8位数据线

START 开始转换信号，上升沿清内部寄存器，下降沿启动A/D转换器

EOC 转换结束信号

ENABLE 输出允许

ADD-A,ADD-B,ADD-C 模拟通道地址线，用以选择端口

ALE 模拟通道地址锁存信号

CLOCK 时钟输入

IN0～IN7 8条模拟信号输入线

Ref(+), Ref(-) 基准电压输入

Vcc，GND 电源端，接地端

实验系统提供的ADC0809其基准电压为5V，模拟通道的地址为地址总线中的低3位，ALE及START信号由地址选中及写信号提供，在产生地址信号时，地址总线的低3位不参加译码，这3位的地址将代表你所选中的模拟通道号。

1.4 可编程并行接口（8255）

8255是一个通用可编程并行接口电路，它具有A、B、C三个8位并行口。其中C口也可用作A、B口的联络信号及中断申请信号。通过编程，它可以被设置为基本输入输出、选通输入输出以及双向传送方式。对于C口还具有按位置0、置1的功能。

8255与CPU之间的连线：

RD、WR 读写信号

RESET 复位信号。

CS 片选信号。

A0～A7 地址信号。

D0～D7 8位数据线。

根据片选端及A0、A1两条地址线可产生4个端口地址，含义如下：

|  |  |
| --- | --- |
| A1 A0 | 端口 |
| 0 0 | A口寄存器 |
| 0 1 | B口寄存器 |
| 1 0 | C口寄存器 |
| 1 1 | 控制口寄存器 |

8255与外部的连线为三个并行口A、B、C的数据总数。当A口或B口定义为选通输入输出及A口定义为双向方式时，C口的某些线为联络信号或中断申请信号。

### 2. 模块分调、联调情况及遇到的困难

2.1 模块分调

分调接线情况见系统总逻辑图。分别调试了6264, 8279, 0809, 8255和打印机，各测试程序出现正常结果就基本就完成了。模块分调中遇到的问题大多由于接线错误引起，出现问题后仔细检查便能得到解决。

2.2 模块联调

模块联调中遇到的问题主要是程序的问题了。分别测试程序各个模块的功能，采用单步和断点并查看寄存器和内存的情况逐个解决即可。

其中遇到的几个解决时间比较长的问题：

1. 程序中漏写offset导致对应内存单元，而不是其地址，被读入bx中等各类问题。

例如一下几处：

mov bx, offset input

最初没有offset就会把input处的指令的二进制值放入bx中，又如：

mtrcwb equ offset mtrcw ;; 正转信号数组边界

mtrcwe equ offset mtrcw + 8

mtrccwb equ offset mtrccw ;; 反转信号数组边界

mtrccwe equ offset mtrccw + 8

mtrtabs equ mtrcwe - mtrcwb ;; 信号数组大小

最初没有offset就会计算数组第一个元素的值，从而出现奇怪的错误。

2. jne等条件跳转指令只能在正负127条指令范围内跳转。但jmp可以在大范围跳转。这是一个编译错误，但通过添加wordptr等修饰仍然无法解决，后来发现条件跳转和非条件跳转的范围是不同的。在一些地方只能先跳转到邻近的地方然后统一用一个jmp跳转到远处。

3. 电机旋转速度不均匀。由于没有采用中断，全部采用轮询的方法来设计程序，而原始程序则在电机旋转一周后才读取一次键盘（即每次读不到键，如果处于在电机功能模块的话，直接调用测试程序里的motor1和motor2函数）。这样的困难在于，当旋转一周后，会有一个不太明显的变速。这是由于每次转了一圈后还需要多执行检查按键等指令的关系，导致最后一个信号与第一个信号的间隔与其他信号不等。解决方法最后采用每次只放出一个信号，然后去测试按键，然后再放出一个信号。这样各个信号的间隔都有按键测试所用的时间，也就很难看出有变速的现象了。（现在想想，觉得应该不会太明显吧，是看错了？还是当初程序的问题？要不就是测试低速旋转的时候感觉出来的，总之，最后就把程序修改成了现在的结构。另外，可能使用中断才是更好的主意）

### 3. 联调流程图及总程序

3.1 联调流程图

调整指针指向  
下一个电机信号  
指针

初始化

读取8279  
键盘

开始

有输入？

Y

处理输入  
设置活动模块

跳转到  
当前活动模块局爱

N

AD0809

读入采样值

输出到  
LED

输出到  
打印机

保存到  
内存

读取PC  
键盘

MOTOR

有输入？

Y

处理输入  
设置转速方向

送出  
电机信号

N

BX = input

BX = ad0809

BX = motor

3.2 联调总程序

data segment

segtab db 3Fh, 06h, 5Bh, 4Fh, 66h, 6Dh ; 0-5

db 7Dh, 07h, 7Fh, 6Fh, 77h, 7Ch ; 6-B

db 39h, 5Eh, 79h, 71h ; C-F

dgtasc db "0123456789ABCDEF" ; 16进制转ASCII码表

spcasc db " " ; 空格

clnasc db ":" ; 分号

;; 正转信号表

mtrcw db 00000001b, 00000011b, 00000010b, 00000110b

db 00000100b, 00001100b, 00001000b, 00001001b

;; 反转信号表

mtrccw db 00001001b, 00001000b, 00001100b, 00000100b

db 00000110b, 00000010b, 00000011b, 00000001b

mtrcwb equ offset mtrcw ;; 正转信号数组边界

mtrcwe equ offset mtrcw + 8

mtrccwb equ offset mtrccw ;; 反转信号数组边界

mtrccwe equ offset mtrccw + 8

mtrtabs equ mtrcwe - mtrcwb ;; 信号数组大小

data ends

;; 各器件端口定义

p0809 equ 08h

padeoc equ 18h

p8279 equ 10h

p8279c equ 11h

p8255a equ 80h

p8255c equ 83h

p8250 equ 180h

p8250c equ 185h

pptr equ 100h

p377 equ 28h

pbusy equ 18h

segext equ 8000h ; 6264的锻号

dlycnt equ 0FFFFh ; 延迟用的循环次数

numspc equ 3 ; 打印两个地址间的空格数

dftspd equ 0F00h ; 电机默认速度

minspd equ 1600h ; 电机最小速度

maxspd equ 0300h ; 电机最大速度

dltspd equ 0100h ; 每次调整的速度

;; 将bx的数值输出到8279上显示

;; 输入：bx

leddsp macro

push ax

mov al, segtab[bx]

not al

out p8279, al

pop ax

endm

;; 将字节$dgt的数值显示在LED上(2位)

;; 输入：$dgt, 不能是bx, cx

leddb macro $dgt

push bx

push cx

mov bx, $dgt

and bx, 00F0h

mov cl, 4

shr bx, cl

leddsp

mov bx, $dgt

and bx, 000Fh

leddsp

pop cx

pop bx

endm

;; 等待打印机可用

;; 占用：al

waitptr macro

local again

again:

in al, pbusy

test al, 80h

jnz again

endm

;; 打印字符$chr

;; 输入：$chr, 不能是al, dx

;; 占用：al, dx

ptrchr macro $chr

waitptr

mov al, $chr

out p377, al

mov dx, pptr

out dx, al

endm

;; 打印bp的低字节

;; 输入：bp

;; 占用：bx, cl

prtbyt macro

mov bx, bp

and bx, 00F0h

mov cl, 4

shr bx, cl

ptrchr dgtasc[bx]

mov bx, bp

and bx, 000Fh

ptrchr dgtasc[bx]

endm

;; 延迟，循环$loops次

delay macro $loops

local again

push cx

mov cx, $loops

again:

loop again

pop cx

endm

;; 栈定义

stack segment

stk db 100 dup(?)

top equ length stk ; 栈大小

stack ends

;; 注：有如下专用寄存器，其在整个程序中有特殊用途，

;; 程序中若需要局部利用，必须在栈中保存其值

;; di 0809采样结果的保存地址指针

;; si 下一个电机信号指针

;; bx 正在执行的功能块地址(指向标签ad0809, motor或者input)

;; cx 电机转速

code segment

assume cs:code, ds:data, ss:stack

main:

mov ax, data

mov ds, ax

mov ax, segext

mov es, ax ; es指向segext(8000)

mov ax, stack

mov ss, ax ; 初始化栈寄存器ss和sp

mov ax, top

mov sp, ax

xor di, di ; di专用：0809结果保存地址指针

mov si, mtrcwb ; si专用：下一个电机信号指针

mov bx, offset input ; bx专用：正在执行的功能块地址

mov cx, dftspd ; cx专用：电机转速

;; 初始化8279

mov al, 11001101b ; 清除显示器ram和fifo

out p8279c, al

mov al, 00101010b ; 设置分频系数为10

out p8279c, al

mov al, 00000000b ; 设置键盘显示器方式

out p8279c, al

mov al, 01010000b ; 设置读fifo/传感器ram模式

out p8279c, al

mov al, 10010000b ; 设置写显示器

out p8279c, al

;; 初始化8255

mov al, 10000000b

out p8255c, al

;; 从8279输入

input:

in al, p8279c

test al, 00000111b

jz run ; 没有输入，执行下一流程(跳转到bx所指的地址)

;; 处理输入

in al, p8279

key1:

cmp al, 0C0h ; key 1

jne key2

mov bx, offset ad0809 ; 设定功能块为ad0809

jmp run

key2:

cmp al, 0C8h ; key 2

jne key3

mov bx, offset motor ; 设定功能块为motor

jmp run

key3:

cmp al, 0D0h ; key 3

jne key4

mov bx, offset input ; 设定不执行任何功能

jmp run

key4:

cmp al, 0D8h ; key 4

je exit ; 程序结束

run:

jmp bx ; 执行bx指定的功能

exit:

int 20h

;; ad0809 功能模块

ad0809:

xor al, al

out p0809, al ; 10H是0809的端口0, 开始采样

waitstart:

in al, padeoc ; 读入0809的eoc

test al, 01h

jnz waitstart ; 如果eoc不为0，则转换尚未启动

waitfinish:

in al, padeoc

test al, 01h

jz waitfinish ; 如果eoc不为1，则转换尚未结束

in al, p0809

;; 显示在LED上

mov dl, al

mov al, 10010000b ; 由于设定有8个LED实际只有4个，故每次设置写显示器

out p8279c, al

leddb di ; 显示地址低2位

leddb dx ; 显示采样数值

;; 输出到打印机

push bx ; 这段代码使用到的寄存器比较多，

push cx ; 故集中保存各个寄存器的值

push dx

push si

mov si, dx ; 采样结果转移到si (dx中间宏里会用到)

mov bp, di ; 地址转移到bp来做取高位、低位的工作

and bp, 0FF00h ; 去高位

mov cl, 8

shr bp, cl

prtbyt ; 打印地址高位

mov bp, di

and bp, 00FFh ; 取低位

prtbyt ; 打印地址低位

prtchr clnasc ; 打印分号

mov bp, si

prtbyt ; 打印采样值(已经转移到si中)

mov cx, numspc ; 打印numspc个空格

spcagn:

prtchr spcasc

loop spcagn

waitptr ; 等待打印结束（应该可以省略的）

pop si

pop dx

pop cx

pop bx

;; 采样值保存到内存

mov es:[di], dl

inc di

delay dlycnt ; 延迟一定时间

jmp input ; 再次读取8279键盘输入

;; 电机模块

motor:

;; 从PC键盘读入并处理按键

mov dx, p8250c

in al, dx

test al, 00000001b

jz mtrrun ; 没有输入，跳转到mtrrun送出一个节拍的电机信号

mov dx, p8250

in al, dx ; 读到输入

keyu:

cmp al, 'u' ; key 'u'

jne keyd

cmp si, mtrcwe

jb mtrrun ; 已经是正转了，无需调整

sub si, mtrtabs ; si指针从正转表跳到反转表的对应位置(避免信号突变)

jmp mtrrun

keyd:

cmp al, 'd' ; key 'd'

jne keyp

cmp si, mtrccwb

jae mtrrun ; 已经是反转了，无需调整

add si, mtrtabs ; si指针从反转表跳到正转表的对应位置(避免信号突变)

jmp mtrrun

keyp:

cmp al, '+' ; key '+'

jne keyeq

keyeq:

cmp al, '=' ; key '=', 也认为是加号，因为shift '=' -> '+'

jne keym

cmp cx, maxspd

jbe mtrrun ; 已经到达最高速度，则什么也不做

sub cx, dltspd ; 加速

jmp mtrrun

keym:

cmp al, '-' ; key '-'

jne mtrrun

cmp cx, minspd

jae mtrrun ; 已经到达最低速度，则什么也不做

add cx, dltspd ; 减速

jmp mtrrun

mtrrun:

mov al, [si]

out p8255a, al ; 输出si所指向的信号

delay cx ; 按照速度(保存在cx)来决定延迟时间

inc si ; 指向下一个位置

cmp si, mtrcwe ; 如果已经到正转达信号表尾端则重置

je rstmtr

cmp si, mtrccwe ; 如果已经到反转达信号表尾端则重置

je rstmtr

jmp input

rstmtr:

sub si, mtrtabs ; 重置：si减去信号表长度即可

jmp input ; 再次读取8279键盘输入

code ends

end main

### 4. 系统总逻辑图和打印结果

6264

A0~A12

D0~D7

WE

CS1

CS2

OE

MEMR

MEMW

SEG 8000

SEG 9000

AD0809

IN 0  
…

IN 7

CS

EOC

CLK

IO 8H

74244 D0

J4

512kHz

8279

RL 0  
…

RL 7

CS

IRQ

CLK

IO 10H

IR 1

1MHz

ROW0~3

SL 0  
…

SL 3

OUTA3

…

OUTB0

COL0~3

L2~L4

A~H

8255

PA 0  
…

PA 3

CS

IO 80H

MOTOR

A, B, C, D

74377

IO 28H

PRINTER

D0~D7

D 0  
…

D 7

CS

74244

IO 18H

D 0  
…

D 7

CS

EOC

BUSY

### 5. 小组分工情况和体会

5.1 分工

shuz负责总程序设计和调试，zg负责连线和系统部件调试。

5.2 体会

通过本次实验，我们实践了关于微机接口和系统间各组件的连接等知识，也进一步巩固了汇编程序设计和调试的技术。