实验二 监控程序使用和汇编语言程序设计实验

1. 实验目的

了解教学机系统的指令格式、寻址方式、基本指令系统构成；

了解机器语言、汇编语言的概念及其对应关系；

学习使用教学机的监控命令操作运行教学及系统的方法和过程；

学习汇编语言程序设计的基础知识和基本技术；

1. 实验说明

这项实验在教学实验设备上完成，更有利于加深对真实教学计算机硬件的了解。为此需要了解教学机系统提供的 A、U、G、D、E、R、T 这 7 个监控命令的格式和每个命令的功能，试着用一下，短时间即可初步了解并学会使用；

汇编语言程序设计则略复杂难一点，需要了解实验设备提供的 30 条基本指令、汇编语 句的格式和各自的功能，以及指令与汇编语句的对应关系。最好先从实验指导书中提供的 程序例子开始，首先学会操作方法和工作步骤；接下来再试着学习、理解每个汇编语句的格式和含义，初步了解汇编程序的结构和设计方法；在有了一些基本知识之后，再开始设计自己的小汇编程序，此时要解决的是把自己想完成的一项事情（例如求整数1到20的累加和）用汇编程序计算出来，还要观察到程序的执行过程及其运行结果。

1. 实验要求

在使用教学机前，应先熟悉教学机的各个组成部分，及其使用方法。

1.学习使用监控命令：

1）用 R 命令查看寄存器内容或修改寄存器的内容。

2）用 D 命令显示存储器内容。

3）用 E 命令修改存储器内容。

4）用 A 命令键入一段汇编源程序，主要是向累加器赋初值并进行运算，执行程查看运行结果。

2.汇编程序设计实验：

1)设计一个小程序, 从键盘上接收一个字符并在屏幕上输出显示该字符。

2)设计一个小程序, 用次数控制在终端屏幕上输出'0'到'9'十个数字符。

3)从键盘上连续打入多个属于‘0’到‘9’的数字符并在屏幕上显示, 遇到非数字字符结束输入过程。

4)计算 1 到 10 的累加和。

5)设计一个有读写内存和子程序调用指令的程序 ,功能是读出内存中的

字符，将其显示到显示器的屏幕上，转换为小写字母后再写回存储器原存储区域。

6)设计一个程序在显示器屏幕上循环显示 95 个（包括空格字符）可打印字符。

1. 连线与设置：



1. 实验内容

操作步骤：

1. 准备一台串口工作良好的PC机；
2. 将TEC-XP放在实验台上，打开实验箱的盖子，确定电源处于断开状态；
3. 将黑色电源线一端接220V交流电源，另一端插在实验箱的电源插座里；
4. 取出通讯线，将通讯线的9芯插头接在实验箱上串口的“COM1”或“COM2”上，另一端接在PC机的串口上；
5. 将实验箱左下方的五个黑色的控制机器运行状态的开关置于正确的位置，在整个实验中开关应置为00110（连续、内存读指令、组合逻辑、联机、16位），开关拨向上方表示“1”，拨向下方表示“0”，“X”表示任意，其它实验中均相同；
6. 打开电源，5V电源指示灯亮；
7. 在PC机上运行PCEC16.EXE文件，根据连接的PC机的串口设置所用PC机的串口为“1”或“2”，其它的设置一般不用改动，直接回车即可。
8. 按一下“RESET”按键，再按一下“START”按键，主机上显示：

TEC-2000 CRT MONITOR

Version 1.0 April 2001

Computer Architecture Lab. , Tsinghua University

Programmed by He Jia

操作过程中的注意事项：

连接电源线和通讯线前实验箱的电源开关一定要处于断开状态，否则可能会对实验仪器或芯片产生损害；

1. 学习使用监控命令
2. 用R命令查看寄存器内容或修改寄存器内容
3. 在命令行提示符状态下输入：

R↙ ；显示寄存器的内容

注：寄存器的内容在运行程序或执行命令后会发生变化。

1. 在命令行提示符状态下输入：

R R0↙ ；修改寄存器 R0 的内容，被修改的寄存器与所赋值之间可以无空格,也可有一个或数个空格

主机显示：

寄存器原值:\_

在该提示下输入新的值0036

再用R命令显示寄存器内容，则R0的值变为0036

1. 用D命令显示存储器内容

在命令行提示符状态下输入：  
D 2000↙

会显示从2000H地址开始的连续128个字的内容；

连续使用不带参数的D命令，起始地址会自动加 128（即80H）。

1. 用E命令修改存储器的内容

在命令行提示符状态下输入：  
E 2000↙

屏幕显示：  
2000 地址单元的原有内容:光标闪烁等待输入一个新的数值

输入 0000

依次改变地址单元2001~2005的内容为:1111 2222 3333 4444 5555

注意：用E命令连续修改内存单元的值时，每修改完一个，按一下空格键，系统会自动给出下一个内存单元的值，等待修改；按回车键则退出E命令。

1. 用D命令显示这几个单元的内容

D 2000↙

可以看到这六个地址单元的内容变为 0000 1111 2222 3333 4444 5555

1. 用A命令键入一段汇编源程序，主要是向累加器赋初值并运算，执行程序查看运行结果。
2. 在命令行提示符状态下输入：

A 2000↙ ；表示该程序从2000H（内存RAM区的起始地址）地址开始

屏幕将显示：2000：

输入如下形式的程序：

2000: MVRD R0，AAAA ；MVRD与R0之间有且只有一个空格，其他指令相同 2002: MVRD R1，5555 ；每条指令都已回车键结束输入

2004: ADD R0，R1

2005: AND R0，R1

2006: RET ；程序的最后一个语句，必须为RET指令

2007:（直接敲回车键，结束 A 命令输入程序的操作过程）

若输入有误，系统会给出提示并显示出错地址，用户只需在该地址重新输入正 确的指令即可。此处的数字均默认为十六进制。

1. 用U命令反汇编刚输入的程序

在命令行提示符状态下输入：

U 2000↙

在从相应地址开始执行反汇编功能，每行一条指令，包括指令地址、指令代码、汇编语句等相应几列的信息。

注：连续使用不带参数的U命令时，将从上一次反汇编的最后一条语句之后接着继续反汇编。

1. 用G命令运行前面键入的源程序

G 2000↙

程序运行结束后，可以看到程序的最终运行结果，屏幕将显示各寄存器的值， 其中R0和R1的值均为 5555H，说明程序运行正确。

1. 用P或T命令，单指令方式执行这段程序，可以查看到每条指令的执行结果

在命令行提示符状态下输入：

T 2000↙

寄存器R0被赋值为 AAAAH

T↙ 寄存器R1被赋值为 5555H

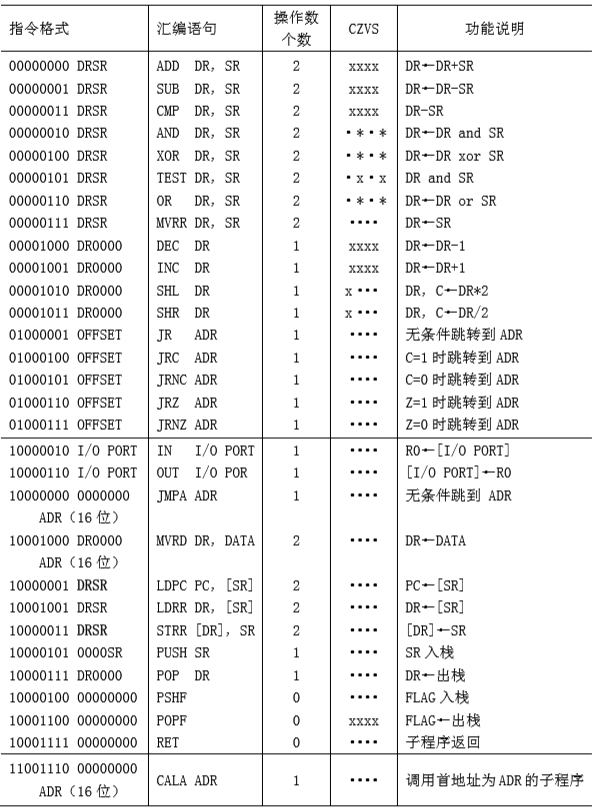
T↙ 做加法运算，和保存进R0,R0的值变为 FFFFH

T↙ 做与运算，结果放在R0,R0的值变为 5555H

注：T总是执行单条指令，但执行P命令时，则把每一个CALL语句连同被调用的子程序一次执行完成。T、P命令每次执行后均显示所有通用寄存器及状态寄存器的内容，并反汇编出下一条即将执行的指令。

1. 汇编程序设计实验

设计汇编语言程序之前，初步看清教学机的基本指令系统的构成，典型指令的功能、指令格式、选用的寻址方式等十分必要，还要对指令代码和汇编语句的对应关系有基本认识，这是本实验要重点学习、深入理解的内容。首先给出教学机的基本指令汇总表。

表1 基本指令汇总表

注：①表中CZVS一列，指令之行后，x表示相关状态位会被重置；·表示不会被修改。 ② 运算器芯片中有16个通用寄存器（累加器）R0~R15，其中：

R4用作堆栈指针SP，其余的用作通用寄存器，IN、OUT 指令默认使用寄存器R0。 用“A”命令输入汇编语言源程序,之后运行并观察执行结果。

实验环境：这项实验选用指令级仿真软件 tec2ksim 系统来进行，有一台装了相应软

件的 PC 机即可。

例1 设计一个小程序，从键盘上接收一个字符并在屏幕上输出显示该字符。

<1> 在命令行提示符状态下输入：

A 2000↙ ；

屏幕将显示：

2000：

输入如下形式的程序：

2000: IN 81 ；判断键盘上是否按了一个键

2001: SHR R0 ；即串行口是否有了输入的字符

2002: SHR R0

2003: JRNC 2000 ；未输入完则循环测试

2004: IN 80 ；接收该字符

2005: OUT 80↙ ；在屏幕上输出显示字符‘6’

2006: RET↙ ；每个用户程序都必须用 RET 指令结束

2007: ↙ ；（按回车键即结束输入过程）

注：在十六位机中，基本 I/O 接口的地址是确定的，数据口的地址为80， 状态口的地址为 81。

<2> 用“G”命令运行程序

在命令行提示符状态下输入：

G 2000↙

执行上面输入的程序

光标闪烁等待输入，用户从键盘键入字符后，屏幕会显示该字符。

该例建立了一个从主存 2000H 地址开始的小程序。在这种方式下,所有的数字都约定使用16进制数,故数字后不用跟字符 H。每个用户程序的最后一个语句一定为RET汇编语句。因为监控程序是选用类似子程序调用方式使实验者的程序投入运行的,用户程序只有用RET语句结束, 才能保证程序运行结束时能正确返回到监控程序的断点, 保证监控程序能继续控制教学机的运行过程。

例2 设计一个小程序，用次数控制在终端屏幕上输出‘0’到‘9’十个数字。

<1> 在命令行提示符状态下输入：

A 2020↙

屏幕将显示：

2020：

从地址2020H开始输入下列程序:

2020: MVRD R2, 000A ；送入输出字符个数

2022: MVRD R0, 0030 ；“0”字符的 ASCII 码送寄存器R0

2024: OUT 80 ；输出保存在R0低位字节的字符

2025: DEC R2 ；输出字符个数减1

2026: JRZ 202E ；判断10个字符输出完否,已完则转程序结束处

2027: PUSH R0 ；未完，保存 R0 的值到堆栈中

2028: IN 81 ；查询接口状态，判断字符串行输出完成否,

2029: SHR R0 ；

202A: JRNC 2028 ；未完成, 则循环等待

202B: POP R0 ；已完成, 从堆栈恢复R0的值，准备继续输出

202C: INC R0 ；得到下一个要输出的字符

202D: JR 2024 ；转去输出字符

202E: RET

202F：↙

该程序的执行码放在2020H起始的连续内存区中。若敲入源码的过程中有错, 系统会进行提示, 等待重新输入正确汇编语句。在输入过程中, 在应输入语句的位置直接打回车则结束输入过程。

<2> 用“G”命令运行程序

在命令行提示符状态下输入：

G 2020↙

执行结果为：

0123456789

思考题: 若把 IN 81, SHR R0, JRNC 2028 语句换成3个 MVRR R0，R0语句, 该程序执行过程会出现什么现象? 试分析并实际执行一次。

类似的, 若要求在终端屏幕上输出'A'到'Z'共26个英文字母，应如何修改例1中给出的程序? 请验证之。

参考答案：

在命令行提示符状态下输入：

A 2100↙

屏幕将显示：2100：

从地址 2100H 开始输入下列程序：

(2100) MVRD R2，001A ；循环次数为26

MVRD R0，0041 ；字符“A”的值

(2104) OUT 80 ；输出保存在R0低位字节的字符

DEC R2 ；输出字符个数减1

JRZ 210E ；判断26个字符输出完否。已完，则转移到程序结束处

PUSH R0 ；未完，保存R0的值到堆栈中

(2108) IN 81 ；查询接口状态，判断字符串行输出完成否

SHR R0 JRNC 2108 ；未完成, 则循环等待

POP R0 ；已完成, 准备输出下一字符，从堆栈恢复R0的值 INC R0 ；得到下一个要输出的字符

JR 2104 ；转去输出字符

(210E) RET

用G命令执行该程序，屏幕上显示“A”~“Z”这26个英文字母。

例3 从键盘上连续输入多个属于‘0’~‘9’的字符并在屏幕上显示，遇到非数字字符结束输入过程。

<1> 在命令行提示符状态下输入：

A 2040↙

屏幕将显示：

2040：

从地址2040H开始输入下列程序:

(2040) MVRD R2, 0030 ； 用于判数字符的下界值

MVRD R3, 0039 ； 用于判数字符的上界值

(2044) IN 81 ； 判断键盘上是否按了一个键

SHR R0 ； 即串行口是否有了输入的字符

SHR R0 JRNC 2044 ； 没有输入则循环测试

IN 80 ； 输入字符到R0

MVRD R1, 00FF

AND R0, R1 ；清零R0的高位字节内容

CMP R0, R2 ； 判断输入字符≥字符'0'否

JRNC 2053 ； 为否, 则转到程序结束处

CMP R3, ； 判输入字符≤字符'9'否

JRNC 2053 ； 为否, 则转到程序结束处

OUT 80 ； 输出刚输入的数字符

JMPA 2044 ；转去程序前边2044处等待输入下一个字符

(2053) RET

<2> 在命令行提示符状态下输入：

G 2040↙

光标闪烁等待键盘输入，若输入0-9十个数字符，则在屏幕上回显；若输入非数字符，则屏幕不再显示该字符，出现命令提示符，等待新命令。

思考题, 本程序中为什么不必判别串行口输出完成否? 设计打入'A'～ 'Z'和'0'～'9'的程序, 遇到其它字符结束输入过程。

例4 计算1到10的累加和。

<1> 在命令行提示符状态下输入：

A 2060↙

屏幕将显示：

2060：

从地址 2060H 开始输入下列程序：

(2060) MVRD R1, 0000 ； 置累加和的初值为0

MVRD R2, 000A ； 最大的加数

MVRD R3, 0000

(2066) INC R3 ； 得到下一个参加累加的数

ADD R1, R3 ； 累加计算

CMP R3, R2 ； 判断是否累加完

JRNZ 2066 ； 未完, 开始下一轮累加

RET

<2> 在命令行提示符状态下输入：

G 2060↙

运行过后, 可以用R命令观察累加器的内容。R1的内容为累加和。 结果为：R1=0037 R2=000A R3=000A

例5 编写程序，把从键盘上输入的小写字母转换为大写字母。

源程序：

code segment

assume cs:code

start:

mov ah,01h

int 21h

mov bl,al

cmp bl,'a'

jb start

cmp bl,'z'

ja start

sub bl,20h

mov dl,bl

mov ah,02h

int 21h

mov ah,4ch

int 21h

code ends

end start

例6 设计一个有读写内存和子程序调用指令的程序，功能是读出内存中的字符，将其显示到显示器的屏幕上，转换为小写字母后再写回存储器的原存储区域。

<1> 将被显示的6个字符 ’A’ ~ ’F’ 送入到内存20F0H开始的存储区域中。

在命令行提示符状态下输入：

E 20F0↙

屏幕将显示： 20F0 内存单元原值：

按下列格式输入：

20F0 内存原值：0041 内存原值：0042 内存原值：0043

内存原值：0044 内存原值：0045 内存原值：0046↙

在命令行提示符状态下输入：

从地址2080H开始输入下列程序：

(2080) MVRD R3, 0006 ；指定被读数据的个数

MVRD R2, 20F0 ；指定被读、写数据内存区首地址

(2084) LDRR R0, [R2] ；读内存中的一个字符到R0寄存器

CALA 2100 ；指定子程序地址为2100,调用子程序，完成显示、转换并写回的功能

DEC R3 ；检查输出的字符个数

JRZ 208B ；完成输出则结束程序的执行过程

INC R2 ；未完成，修改内存地址

JR 2084 ；转移到程序的2086处，循环执行规定的处理

(208B) RET

从地址2100H开始输入下列程序：

(2100) OUT 80 ；输出保存在R0寄存器中的字符

MVRD R1, 0020

ADD R0, R1 ；将保存在R0中的大写字母转换为小写字母

STRR [R2],R0 ；写R0中的字符到内存，地址同LOD所用的地址

(2105) IN 81 ；测试串行接口是否完成输出过程

SHR R0 JRNC 2105 ；未完成输出过程则循环测试

RET ；结束子程序执行过程，返回主程序

<2> 在命令行提示符状态下输入：

G 2080↙

屏幕显示运行结果为：

ABCDEF

<3> 在命令行提示符状态下输入：

D 20F0↙

20F0H~20F5H 内存单元的内容为： 0061 0062 0063 0064 0065 0066

例6 设计一个程序在显示器屏幕上循环显示95个（包括空格字符）可打印字符。

<1>在命令行提示符状态下输入：

A 20A0↙

屏幕将显示：

20A0：

从地址 20A0H 开始输入下列程序：

A 20A0 ；从内存的20A0单元开始建立用户的第一个程序

20A0: MVRD R1, 7E ；向寄存器传送立即数

20A2: MVRD R0, 20 ；

20A4: OUT 80 ；通过串行接口输出R0低位字节内容到显示器屏幕

20A5: PUSH R0 ；保存R0寄存器的内容到堆栈中

20A6: IN 81 ；读串行接口的状态寄存器的内容

20A7: SHR R0 ；R0的内容右移一位，最低位的值移入标志位C

20A8: JRNC 20A6 ；条件转移指令，当标志位C不是1时就转到20A6地址

20A9: POP R0 ；从堆栈中恢复R0寄存器的原内容

20AA: CMP R0, R1 ；比较两个寄存器的内容是否相同，相同则标志位Z=1

20AB: JRZ 20A0 ；条件转移指令，当标志位Z为1时转到20A0地址

20AC: INC R0 ；把R0寄存器的内容增加1

20AE: JR 20A4 ；无条件转移指令，一定转移到20A4地址

20AF: RET ；子程序返回指令,程序结束

<2> 在命令行提示符状态下输入：

G 20A0↙

运行过后, 可以观察到显示器上会显示出所有可打印的字符。

上述例子，都是用监控程序的A命令完成输入源汇编程序的。在涉及到汇编语句标号 的地方，不能用符号表示，只能在指令中使用绝对地址。使用内存中的数据，也由程序员给出数据在内存中的绝对地址。显而易见，对这样的很短小程序矛盾并不突出，但很容易想到，对很大的程序，一定会遇到相当大的困难。

例7从键盘上接收两个一位十六进制数据，求其和，并以十进制形式显示出来。

源代码：

code segment

assume cs:code

start:

mov cx,02h ;loop循环次数

mov bl,00h

input:

mov ah,01h

int 21h

sub al,30h ;数字

cmp al,0ah

jl next

sub al,07h ;大写字母

cmp al,10h

jl next

sub al,20h ;小写字母

next:

add bl,al ;把两数之和放在bh中

loop input ;转化为十进制数

mov al,bl

cbw

cmp al,0ah

jg greater

mov dl,al

add dl,30h

jmp out1

greater:

mov cl,0ah

idiv cl

mov cl ,06h

imul cl

add bl,al ;十进制显示

mov al,bl

mov cl,04h

shr al,cl

and al,0fh

jz jump

mov dl,al

add dl,30h

mov ah,02h

int 21h

jump:

and bl,0fh

mov dl,bl

add dl,30h

out1:

mov ah,02h

int 21h

mov ah,4ch

int 21h

code ends

end start