

# 实验四 时钟实验

学号：61822313

姓名：钟锦程

实验日期：2024 年 4 月 29 日

## 1 实验任务和实验结果

### 1.1 基础实验任务

执行时钟程序时，屏幕上显示提示符':', 由键盘输入当前时、分和秒值，即 XX:XX:XX，随即显示时间，并不停地计时。当有键按下时，立即停止计时，返回 DOS。

#### 1.1.1 调试通过的源程序

```
1 DATA SEGMENT
2     TIME DB 9 ;存放时间ASCII码的缓冲区
3         DB ?
4         DB 9 DUP(0)
5     D2 DB 0DH,0AH,'$';0D是光标移到行首，0A是换行
6     D3 DB 0AH,'$'
7 DATA ENDS
8 CODE SEGMENT
9     ASSUME DS:DATA,CS:CODE
10 START:
11     MOV AX,DATA
12     MOV DS,AX
13     MOV AH,2;显示冒号
14     MOV DL,':'
15     INT 21H
16     LEA DX,TIME;让用户输入时间XX:XX:XX
17     MOV AH,0AH
18     INT 21H
19     MOV SI,OFFSET TIME ;ASCII TO BCD
20     MOV AL,0
21     MOV [SI+10],AL ;删掉缓冲区其余部分的内容
22     MOV DX,OFFSET D3 ;将光标移到下一行，否则第一次输出会覆盖之前的内容
23     MOV AH,9
```

```
24     INT 21H
25 NEXT:
26     MOV CH, [SI+2]; HOUR(ASCII) TO BCD 存放在CH
27     AND CH, 0FH
28     MOV CL, 4
29     SHL CH, CL
30     MOV BL, [SI+3]
31     AND BL, 0FH
32     ADD CH, BL
33     MOV DH, [SI+5]; MINUTE(ASCII) TO BCD 存放在DH
34     AND DH, 0FH
35     SHL DH, CL
36     MOV BL, [SI+6]
37     AND BL, 0FH
38     ADD DH, BL
39     MOV DL, [SI+8]; SECEND(ASCII) TO BCD 存放在DL
40     AND DL, 0FH
41     SHL DL, CL
42     MOV BL, [SI+9]
43     AND BL, 0FH
44     ADD DL, BL
45     CALL DELAY
46     MOV AL, DL ;秒值增1
47     ADD AL, 1
48     DAA
49     MOV DL, AL
50     CMP DL, 60H;秒值与60比较, 如果不等于则直接显示
51     JNZ SHOW
52     MOV DL, 00H ;秒置零
53     MOV AL, DH
54     ADD AL, 1 ;分钟+1
55     DAA
56     MOV DH, AL
57     CMP DH, 60H;分钟值与60比较, 如果不等于则直接显示
58     JNZ SHOW
59     MOV DH, 00H;分钟置零
60     MOV AL, CH;小时+1
61     ADD AL, 1
62     DAA
```

```
63     MOV CH,AL
64     CMP CH,24H ;小时值与24比较, 如果不等于则直接显示
65     JNZ SHOW
66     MOV CH,00H;若小时值等于24, 则置零后再转显示
67     JMP SHOW
68 SHOW:
69     MOV CL,4
70     MOV AX,0000H;小时从BCD转换为ASCII并存到缓冲区
71     MOV AL,CH
72     SHL AX,CL
73     SHR AL,CL
74     OR AX,3030H
75     XCHG AH,AL
76     MOV [SI+2],AX
77     MOV AX,0000H;分钟从BCD转换为ASCII并存到缓冲区
78     MOV AL,DH
79     SHL AX,CL
80     SHR AL,CL
81     OR AX,3030H
82     XCHG AH,AL
83     MOV [SI+5],AX
84     MOV AX,0000H;秒从BCD转换为ASCII并存到缓冲区
85     MOV AL,DL
86     SHL AX,CL
87     SHR AL,CL
88     OR AX,3030H
89     XCHG AH,AL
90     MOV [SI+8],AX
91     MOV DX,OFFSET TIME;输出到屏幕
92     ADD DX,2;指向小时值开头的地址
93     MOV AH,9
94     INT 21H
95     MOV AH,06;检查是否有键按下
96     MOV DL,0FFH
97     INT 21H
98     JNZ EXIT
99     JMP NEXT
100 EXIT:
101     MOV AH,4CH ;若有键按下, 结束程序
```

```
102     INT 21H
103
104 DELAY PROC
105     PUSH CX;保护可能改变的寄存器
106     PUSH DX
107     PUSH AX
108     MOV AH,2CH;读取系统时间, DH得到秒值
109     INT 21H
110     MOV AL,DH
111     ADD AL,1;手动+1, 使AL表示下一秒
112 READTIME:
113     MOV AH,2CH;读取时间DH得到秒
114     INT 21H
115     CMP DH,AL
116     JNZ READTIME;若不等于下一秒, 则再读取时间;若相等, 则结束延时
117     POP AX
118     POP DX
119     POP CX
120     RET
121 DELAY ENDP
122 CODE ENDS
123     END START
```

### 1.1.2 实验结果

如图1是刚打开程序时的显示, 屏幕打印了一个冒号, 并等待键盘输入。图2是输入时间 23:59:55 后的显示。截图第一排是输入的起始时间, 自输入后, 每隔一秒时间值加一秒并显示一次。在程序正常运行期间随机按下一个键, 程序终止。截图最后一排显示程序已经停止, 询问是否要留在 DOSBox 中。

```

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: EXP4_1
HAVE FUN!
The DOSBox Team http://www.dosbox.com
Z:\>SET BLASTER=A220 I7 D1 H5 T6
Z:\>mount c "c:\Users\Leo\AppData\Roaming\Code\User\globalStorage\xsro.masm-tasm\MM5M-v6.11"
Drive C is mounted as local directory c:\Users\Leo\AppData\Roaming\Code\User\globalStorage\xsro.masm-tasm\MM5M-v6.11\
Z:\>mount d "d:\LEODOS"
Drive D is mounted as local directory d:\LEODOS\
Z:\>d:
D:\>set PATH=D:\LEODOS
D:\>masm D:\exp4_1.asm; >>C:\Z7429.LOG
D:\>link D:\exp4_1; >>C:\Z7429.LOG
Warning: no stack segment
D:\>D:\exp4_1

```

图 1: 基础实验任务结果截图 1

```

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX
23:59:55
23:59:56
23:59:57
23:59:58
23:59:59
00:00:00
00:00:01
00:00:02
00:00:03
00:00:04
00:00:05
00:00:06
00:00:07
00:00:08
00:00:09
00:00:10
00:00:11
00:00:12
00:00:13
00:00:14
00:00:15
00:00:16
00:00:17
Do you need to keep the DOSBox (Y,N)?

```

图 2: 基础实验任务结果截图 2

## 1.2 附加实验任务

1. 在同一行的相同位置显示更新的计时时间，不换行。
2. 输入时间初值时，能够检查是否存在错误、提示错误信息，并可重新输入时间初值。错误类型及提示信息分为两种：1. 输入的时间初值是错误的字符，即不是数字或冒号；2. 输入的时间值是错误的，即“时”大于等于 24，“分”和“秒”大于等于 60。
3. 延时一秒采用 DOS 系统功能调用实现

### 1.2.1 调试通过的源程序

```

1 DATA SEGMENT
2     TIME DB 9
3         DB ?
4         DB 9 DUP(0)
5     D2 DB 0DH, '$'; 0D是光标移到行首, 0A是换行
6     D3 DB 0AH, '$'
7     ERRINFO1 DB 0AH, 'WRONG INFO 1: INPUT NOT NUMBER OR COLON', 0DH, 0AH, '$'
8     ERRINFO2 DB 0AH, 'WRONG INFO 2: INPUT TIME UNREASONABLE', 0DH, 0AH, '$'
9     RETYPEINFO DB 'INPUT AGAIN:', 0DH, 0AH, '$'
10 DATA ENDS
11 CODE SEGMENT
12     ASSUME DS:DATA, CS:CODE
13 START:
14     MOV AX, DATA
15     MOV DS, AX
16     MOV AH, 2; 显示冒号

```

```
17     MOV DL, ':'
18     INT 21H
19     MOV DX, OFFSET D3
20     MOV AH, 9
21     INT 21H
22     LEA DX, TIME;让用户输入时间XX:XX:XX
23     MOV AH, 0AH
24     INT 21H
25     MOV SI, OFFSET TIME
26     MOV AL, 0
27     MOV [SI+10], AL
28     CALL CHECK ;调用检查子程序。检查输入是否错误，提示错误信息，若错误，重新输入
        时间初值
29 NEXT:
30     CALL ATOB;调用ASCII到BCD码的转换子程序
31     CALL DELAY ;调用延时程序
32     MOV AL, DL ;秒值增1
33     ADD AL, 1
34     DAA
35     MOV DL, AL
36     CMP DL, 60H
37     JNZ SHOW
38     MOV DL, 00H ;秒置零
39     MOV AL, DH
40     ADD AL, 1 ;分钟+1
41     DAA
42     MOV DH, AL
43     CMP DH, 60H
44     JNZ SHOW
45     MOV DH, 00H;分钟置零
46     MOV AL, CH;小时+1
47     ADD AL, 1
48     DAA
49     MOV CH, AL
50     CMP CH, 24H
51     JNZ SHOW
52     MOV CH, 00H
53     JMP SHOW
54 SHOW:
```

```
55     MOV CL,4
56     MOV AX,0000H;小时从BCD转换为ASCII并存到缓冲区
57     MOV AL,CH
58     SHL AX,CL
59     SHR AL,CL
60     OR AX,3030H
61     XCHG AH,AL
62     MOV [SI+2],AX
63     MOV AX,0000H;分钟从BCD转换为ASCII并存到缓冲区
64     MOV AL,DH
65     SHL AX,CL
66     SHR AL,CL
67     OR AX,3030H
68     XCHG AH,AL
69     MOV [SI+5],AX
70     MOV AX,0000H;秒从BCD转换为ASCII并存到缓冲区
71     MOV AL,DL
72     SHL AX,CL
73     SHR AL,CL
74     OR AX,3030H
75     XCHG AH,AL
76     MOV [SI+8],AX
77     MOV DX,OFFSET TIME;输出到屏幕
78     ADD DX,2
79     MOV AH,9
80     INT 21H
81     MOV AH,06;检查是否有键按下
82     MOV DL,0FFH
83     INT 21H
84     JNZ EXIT
85     JMP NEXT
86 EXIT:
87     MOV AH,4CH ;若有键按下, 结束程序
88     INT 21H
89
90 ATOB PROC
91     MOV CH,[SI+2];HOUR(ASCII)TO BCD 存放在CH
92     AND CH,0FH
93     MOV CL,4
```

```
94     SHL CH,CL
95     MOV BL,[SI+3]
96     AND BL,0FH
97     ADD CH,BL
98     MOV DH,[SI+5];MINUTE(ASCII)TO BCD 存放在DH
99     AND DH,0FH
100    SHL DH,CL
101    MOV BL,[SI+6]
102    AND BL,0FH
103    ADD DH,BL
104    MOV DL,[SI+8];SECEND(ASCII)TO BCD 存放在DL
105    AND DL,0FH
106    SHL DL,CL
107    MOV BL,[SI+9]
108    AND BL,0FH
109    ADD DL,BL
110    RET ;不加RET会继续执行下面的DELAY, 最终导致每次延时多一秒!
111    ATOB ENDP
112 DELAY PROC;延时子程序
113     PUSH CX
114     PUSH DX
115     PUSH AX
116     MOV AH,2CH;读取时间DH得到秒
117     INT 21H
118     MOV AL,DH
119     ADD AL,1;AL表示下一秒
120 READTIME:
121     MOV AH,2CH;读取时间DH得到秒
122     INT 21H
123     CMP DH,AL
124     JNZ READTIME;若不等于下一秒, 则再读取时间;若相等, 则结束延时
125     POP AX
126     POP DX
127     POP CX
128     RET
129 DELAY ENDP
130 CHECK PROC ;检查输入的子程序
131     ;错误信息1: 输入的时间不是数字或冒号
132     ;错误信息2: 输入的时间不符合实际
```



```

133 ;此时SI已经指向TIME首址
134
135 ;第一步：用ASCII码检查是不是数字或冒号
136 PUSH CX;保护可能改变的寄存器
137 PUSH AX
138 PUSH DX
139 PUSH BX
140 MOV CX,8;置循环次数
141 MOV BX,SI;把缓冲区首址复制给BX
142 ADD BX,1
143 AGIN:
144 MOV AX,SI
145 ADD AX,4;AX用来指示冒号的位置
146 INC BX;第一次循环开始时，BX(=SI+2)指到第一个字符，BX始终指向想要比较的字符
147 CMP BX,AX;SI+4和SI+7指到冒号
148 JZ CHECKCOLON;若指针指到冒号，则跳转冒号检查子程序
149 ADD AX,3;指向下一个冒号
150 CMP BX,AX
151 JZ CHECKCOLON ;若指针指到冒号，则跳转冒号检查子程序
152 JMP CHECKNUM ;两次都没有判定为指到冒号，说明是数字，跳转数字检查子程序
153 CHECKCOLON:;冒号检查
154 CMP BYTE PTR [BX],':'
155 JNZ WRONG1
156 LOOP AGIN
157 CHECKNUM:;数字检查
158 ;AX不能用来存放偏移地址!!!
159 CMP BYTE PTR [BX],30H
160 JB WRONG1
161 CMP BYTE PTR [BX],39H
162 JA WRONG1
163 LOOP AGIN
164 ;第二步：检查输入时间是否合理
165 CALL ATOB ;调用ASCII到BCD的转换子程序。时分秒信息已经转换为BCD放到CH,
    DH,DL
166 CMP CH,23H ;若小时数大于23，说明不合理，输出错误信息2
167 JA WRONG2
168 CMP DH,59H;若分钟数大于59，说明不合理，输出错误信息2
169 JA WRONG2

```

```

170     CMP DL,59H;若秒数大于59,说明不合理,输出错误信息2
171     JA  WRONG2
172     JMP  OVER ;若不加这一句,无论如何都会执行WRONG1!!!
173  WRONG1:;输出错误信息1: 输入的时间不是数字或冒号
174     MOV DX,OFFSET ERRINF01
175     MOV AH,9
176     INT 21H
177     JMP RETYPE
178  WRONG2:;输出错误信息2: 输入的时间不符合实际
179     MOV DX,OFFSET ERRINF02
180     MOV AH,9
181     INT 21H
182     JMP RETYPE
183  RETYPE:;提示重新输入并接收新的输入
184     MOV DX,OFFSET RETYPEINFO
185     MOV AH,9
186     INT 21H
187     LEA DX,TIME;让用户输入时间XX:XX:XX
188     MOV AH,0AH
189     INT 21H
190     CALL CHECK
191  OVER:
192     POP BX ;恢复现场
193     POP DX
194     POP AX
195     POP CX
196     RET
197     CHECK ENDP
198  CODE ENDS
199     END START

```

### 1.2.2 实验结果

图3是两种错误输入的结果。第一种错误输入：输入的不是数字或冒号。图中第一次在本应该输入数字的地方输入了字母 a，程序显示了第一类错误信息并提示重新输入；第二次在本应该输入冒号的地方输入了分号，程序也显示了第一类错误信息并提示重新输入。第三次输入了一个不合理的时间值，即小时数大于 23，程序输出了第二类错误信息并提示重新输入。图4是正确规范的输入。输入后，每隔一秒在原位置更新时间值（没有换行），图5显示了更新过程中的一个瞬间，表明时钟可以正常运行。最后，图6显示：在程序运行过程中，在键盘上

随便按一个键，可以让程序终止。

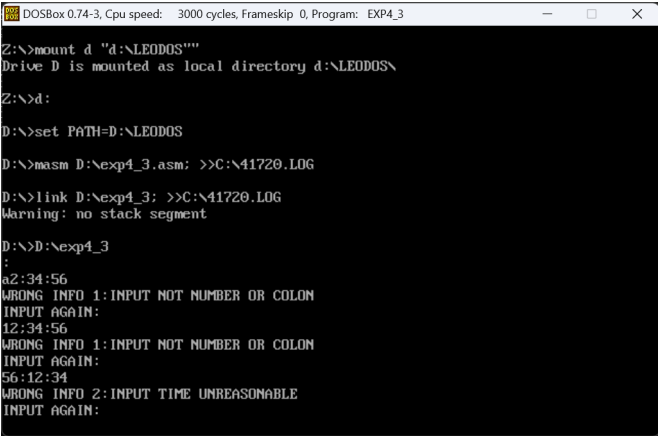


图 3: 附加实验任务结果截图 1

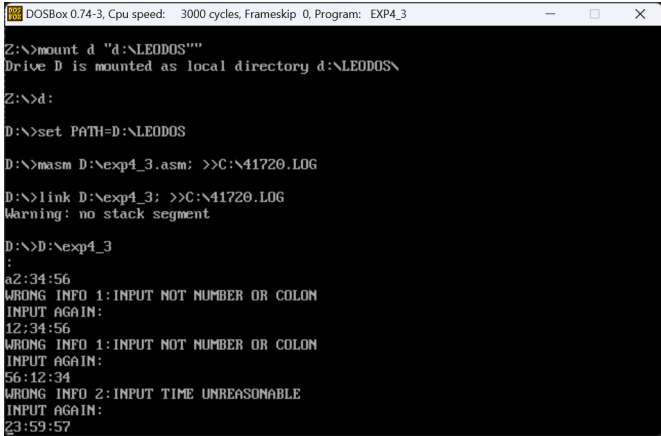


图 4: 附加实验任务结果截图 2

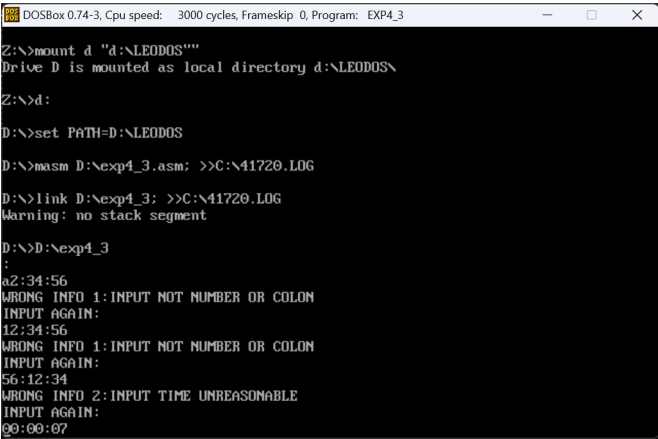


图 5: 附加实验任务结果截图 3

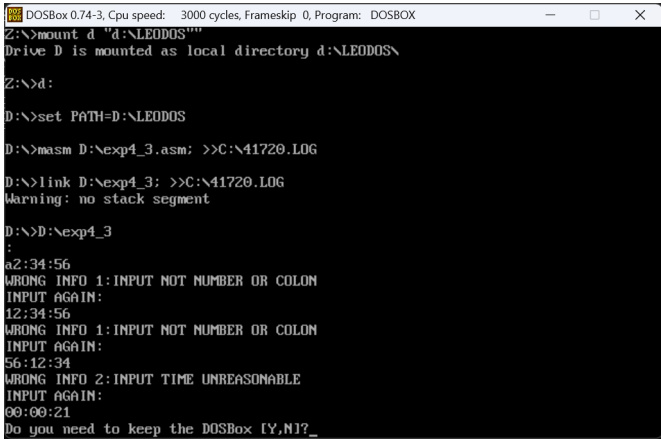


图 6: 附加实验任务结果截图 4

## 2 实验总结

在完成基础任务时，遇到以下问题：

1. 没有理解 0DH 和 0AH 的作用，光标位置不在预想的位置。0DH 使光标移到行首，0AH 换行。要输出更新后的时间之前，应该先换行，否则会覆盖之前的输出。
2. 编写子程序的时候，最初没有保护现场，导致寄存器的值被改变，程序崩溃。解决方法是：编写子程序时先将可能改变的寄存器入栈，子程序结束后再出栈恢复现场。
3. 编写延时程序时，最开始有一步设置了系统时间，后来发现这一步是不必要的，且系统时间也不应该修改，遂删去。

在完成附加任务时，遇到以下问题：

1. 为了增加代码复用性，将 ASCII 码到 BCD 码的转换写成子程序 (ATOB)，但是在结尾处忘了写 RET，结果每次调用 ATOB 后程序会顺次执行下面的 DELAY 子程序，导致每次延时都会多 1 秒。加上 RET 后恢复正常。
2. 编写检查子程序 (CHECK) 时，最初用到 AX 来做指针指向每一个待检查的字符，但发现编译结果总与预想不同。上网查阅发现 AX, CX, DX 均不能存放地址。于是改用 BX，使问题得到解决。
3. 编写检查程序时，发现即使输入正确也会打印第一类错误信息。经检查，是因为在经过所有检查流程（即输入正确）后，没有加跳转到子程序出口的指令。而此处后面正好跟的是输出第一类错误信息的程序，所以会出现错误。解决办法：加上 JMP OVER，使得经过所有检查流程直接跳转到子程序出口。

### 3 思考题

时钟程序中存在时间误差吗？若有误差，其来源在何处？如何进行误差校正？

答：存在误差。来源于程序本身运行的耗时和延时程序的误差。其中后者是主要误差来源。本程序中，延时子程序的逻辑是：读取系统时间，将秒值加 1，再次读取系统时间，并将新读出的秒值与加 1 后的结果比较，如果不同，则再次读取如此循环，如果相同，说明距离第一次读取（即进入延时程序的瞬间）已经过去一秒，这时退出延时子程序。这种延时可能会在首次调用该子程序时有较大误差。

这种延时的缺陷在于，仅检查了秒值。忽略计算机执行耗时，假若第一次读取系统时间为 00:00:00 50ms，则在 00:00:01 00ms 瞬间就会退出延时，延时不足 1 秒。若考虑计算机执行的耗时，则会出现新的误差。在延时程序以外，不能保证每次其他部分的程序都执行 1 秒的整数倍。否则由于同样的原因，除了首次调用，后续每一次调用都会产生误差。改进办法是同时检查秒值与毫秒值，仅当秒值与初始秒值加一后的结果相同且毫秒值也相同的时候才退出。